Buyer's Guide To The Latest Printers

COMPUTE!

$3.00
June
1986
Issue 73
Vol. 8, No. 6
$3.95 Canada
02193
ISSN 0194-357X

The Leading Magazine Of Home, Educational, And Recreational Computing

Atari ST Hints & Tips
How To Customize Icons, Autorun Programs, And More

Miami Ice
A Slick Action Game For Commodore 64, 128, Atari, And Apple

Looking Glass
Add Window Commands To The Commodore 64

Atari Password
Protect Your Programs From Prying Eyes

Automatic Typist
Using Exec Files On The Apple II

Meet ED:
The AmigaDOS Text Editor

Minding Memory From IBM BASIC
Put An End To RAM Wars

06
71486 02193


MichTron
DIGITAL RESEARCH
57904 bytes used in 8 it
BATTERIES INCLUDED
EPYX
MIGRAPH
SIERRA
Brute
HP....363
ATARI SC1224
ACTIVISION
VIVACE
subLOGIC
SPINNAKER
and in walks Della,
smiling mysteriously.
"Looks to me like you had
yourself a female visitor
last night, Perry.
Entertaining a new
client?"
INFOCOM
You have:
no tea
a loose pile of junk mail
a toothbrush
a flathead screwdriver
your gown (being worn)
It looks like your gown contains:
pocket fluff
The vast yellow ships thunder across the sky, spreading waves of terror and
panic in their wake. The voice of the Vogon Captain slams across the country,
the local planning office in Alpha Centauri for fifty years and it's too late to
start making a fuss about it now.

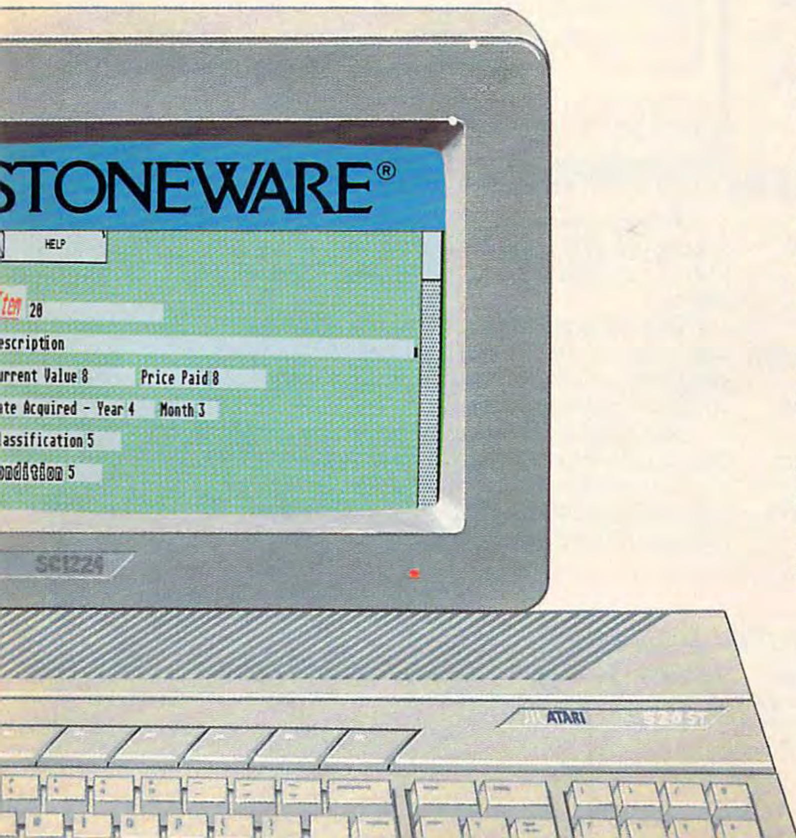
THE ST COMPUTER LINE FROM ATARI.

IT'S ALREADY KNOWN BY THE COMPANY IT KEEPS.

It's only natural that the hottest new computers in America are attracting the hottest software companies in the business.

The library of innovative business, education, entertainment, system management, and integrated package software for the 520ST™ and the new 1040ST™ is already impressive, with literally dozens of new programs being introduced almost every week.

In fact, the software companies who are committing their time, money, and expertise to the ST are the same companies who regularly show up on all the software hit lists.



And it's no wonder that the leading software developers are excited by the power and speed of the ST Computers.

Stoneware®, for example, checked out the speed of the ST Disk Drive in data base applications and flipped. Instead of having to wait forever to manipulate data, thousands of records can be sorted in a fraction of the time that it takes on other computers. And instant responsiveness is the name of the game, not waiting.

Sierra On-Line®, on the other hand, took one look at our incredible high speed, high resolution graphics and was ecstatic. The result is a whole series of games that are more realistic and lifelike than ever before.

For their needs, Activision® focused on the built-in MIDI ports for attaching synthesizers and other musical instruments. This enabled them to design the ultimate program for playing and composing music.

The list goes on and on. But a designer for Spinnaker® perhaps summed up the capabilities of the 520ST and the new 1040ST best:

"I feel like a painter," he said, "who at last has a canvas large enough to let my creativity run free."

So if you're looking for a computer system that combines the very best in software with the very best in hardware at the very best price . . . you're looking for an ST from Atari.

ATARI®


THOMSON

HOW DO YOU TOP APPLE, COMMODORE, OR IBM BEAUTIFULLY?

SEE YOUR THOMSON DEALER.

Alabama
Mel's Photo Shop, Montrose
Mall Ted, Mobile
Conrad Computer, Birmingham
Command Computer, Birmingham

Alaska
Fred Meyer Photo Electronic Centers:
Juneau, Anchorage, Fairbanks
Hy-Tech Systems, Juneau

Arizona
Compu Save, Phoenix
National Business Software, Tempe

California
Bill's TV, Santa Clara
P.C. Edge, Santa Clara
Grand Central Software, Watsonville
Los Altos PC, Los Altos
Davis Office Systems, Merced
Computer Biz: San Jose, Cupertino
Computer Nook, San Bernardino
Personal Computer Service,
Marina del Rey

Colorado
Software Etc., Denver
Computer Dynamics, Colorado Springs
Computer Interface, Golden
MSR Inc., Denver
Classic Computers, Fort Collins
CRT, Fort Collins
Computer Source, Boulder
Alpha Center, Denver

Connecticut
Crazy Eddie's, All Locations

Florida
Random Access, Ft. Walton Beach
Office World, Sarasota
AA Computer, Jacksonville
Computer Base, Tampa
Discount Disc, Shalimar
G.C.C.E., Panama

Hawaii
Computer House, Honolulu
PC Price Busters, Honolulu

Idaho
Fred Meyer Photo Electronic Centers:
Boise, Pocatello, Idaho Falls

Illinois
Aardvark Computer, Chicago Heights
Protecto, Barrington

Kansas
Computer Center Inc., Olathe

Minnesota
Computer Exercise World, Minneapolis

Mississippi
Enterprise Unlimited, Jackson

Missouri
Micronomics, Winchester

Nevada
Computer House, Sparks

New Jersey
Computer Software Sales, Ocean
Super Software, E. Brunswick
Crazy Eddie's, All Locations

New Mexico
Suntec Computers, Albuquerque
Omega Business Products, Albuquerque

New York
Sysut Computers, Brooklyn
Crazy Eddie's, All Locations

North Carolina
Triad Computer, Greensboro
Hamilton Computer, Greensboro
Chips Computer, Fayetteville
Two M. Systems, Jacksonville

North Dakota
Computerland, Fargo

Ohio
Microwave Magic, Fairfield
Electronic Connection, Centerville
Fairborn Home Computers, Fairborn
Quality Computer Application, Toledo

Oklahoma
Master Systems Ltd., Bethany

Oregon
Schad's Electronics, Eugene
Fred Meyer Electronic Centers:
Portland, Albany, Beaverton,
Coos Bay, Salem, Gresham, Medford,
Roseburg, Eugene

Pennsylvania
Data Softique Computer, Pittsburgh

South Carolina
S.F. Enterprises, Sandy Springs
Bytes & Bits, Columbia
Con Current Technology, Travelers Rest
Software Haus, Charleston Heights
Micro Computer Depot, Sumpter

Tennessee
Electronic Warehouse, Decherd
Data World, Knoxville
Computers International, Hixson
Computer King, Knoxville

Texas
Permian Micromart, Odessa
Turnkey Computer Systems, Houston

Utah
Fred Meyer Electronic Centers:
Orem, Ogden, Logan, Salt Lake City
Digital World, West Valley City
Burgoyne Computer Inc.,
South Salt Lake City
Research Micro Systems, Provo

Virginia
Shelken Associates, Inc.,
Fredericksburg

Washington
Fred Meyer Photo Electronic Centers:
Seattle, Tacoma, Bellevue,
Bellingham, Longview

Wisconsin
Micro Age, Milwaukee
Inacomp, Madison

Wyoming
Cheyenne Computer, Cheyenne

THOMSON
A SIGHT FOR SORE EYES.™

Corporate Address
5731 West Slauson Avenue, Suite 111, Culver City, CA 90230
(213) 568-1002
(800) 325-0464 (Outside CA)

Children's Software from Springboard.™

The best way to introduce your child to the computer.

Early Games for Young Children.™

This popular program offers a delightful collection of games designed to teach children ages 2½ to 6 a variety of important basic skills in a fun and friendly way.

Children learn to distinguish shapes, recognize letters, say the alphabet, count, add, subtract, even spell their names!

Large, colorful letters, gentle hints and a picture menu help children learn without adult supervision.

It's the perfect way to introduce your child to the computer.

Early Games

Satisfaction Guaranteed

At Springboard, we stand behind our products.

That's why we offer a full 30-day money-back guarantee. Look for these and other Springboard products for all ages at your favorite software store.

Easy as ABC

Now Available For:

Macintosh

Apple II+, IIe, IIc, IBM PC, PCjr, Commodore 64/128

Easy as ABC.™

Learning the alphabet is a wonderful new adventure with Easy as ABC. Five fun games introduce children ages 3 to 6 to letter recognition, alphabetical sequence and upper/lower case letters.

Delightful animals present the alphabet in a variety of ways. Jumping frogs, buzzing bees and soaring rockets fascinate your children as they learn. And multiple skill levels keep them interested so they learn more.

No other program teaches your children the alphabet as thoroughly.

SPRINGBOARD

Springboard Software, Inc. • 7808 Creekridge Circle • Minneapolis, MN 55435 • (612) 944-3915

COMPUTE!
JUNE 1986
VOLUME 8
NUMBER 6
ISSUE 73

NOTE: See page 117 before typing in programs.

AP Apple, **Mac** Macintosh, **AT** Atari, **ST**, Atari ST, **V** VIC-20, **64** Commodore 64, **+4** Commodore Plus/4, **16** Commodore 16, **128** Commodore 128, **P** PET/CBM, **TI** Texas Instruments, **PC** IBM PC, **PCjr** IBM PCjr, **AM** Amiga. *General interest.

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
ABC Publishing, President, Robert G. Burton
1330 Avenue of the Americas, New York, New York 10019

COMPUTE! The Journal for Progressive Computing (USPS: 537250) is published monthly by COMPUTE! Publications, Inc., 825 7th Ave., New York, NY 10019 USA. Phone: (212) 265-8360. Editorial Offices are located at 324 West Wendover Avenue, Greensboro, NC 27408. Domestic Subscriptions: 12 issues, $24. POSTMASTER: Send address changes to: **COMPUTE!** Magazine, P.O. Box 10955, Des Moines, IA 50950. Second class postage paid at Greensboro, NC 27403 and additional mailing offices. ISSN 0194-357X.

Editor's Notes

A New Challenge For ST Programmers

If you're an Atari ST enthusiast, we've got some fantastic news for you.

There's only one catch. We can't tell you what the news is—yet. All we can say is that COMPUTE! is preparing a major surprise that we think you'll enjoy. And to make this surprise as fantastic as possible, we need your help.

If you have access to an ST, and if you're a skillful programmer or writer, we want to see your work. We're looking for ST-oriented articles on a wide variety of topics: tutorials, application programs, utilities, games, educational programs, or almost anything else that we think will be of interest to the several hundred thousand people who already own and use Atari ST computers.

And to break things really wide open, in this instance we're modifying a longstanding COMPUTE! policy regarding program submissions. Up to now, for the most part, we've restricted the programs we publish to either BASIC or machine language, and we've also restricted their length. This has forced us to turn down some otherwise outstanding submissions, but we've had good reasons for it.

These reasons have to do with the realities of magazine publishing. We've insisted that programs be written in BASIC or ML because those are the only two languages that everybody owns. Practically every personal computer comes with BASIC, and ML is every computer's native language. If we publish a program written in some other language—such as Pascal, C, COMAL, Forth, or whatever—the number of readers who can use the program suddenly shrinks to a tiny minority. Realistically, a magazine that wants to stay in business has to appeal to a majority of its readers most of the time. Thus, we've avoided programs written in "nonstandard" languages, although it's been frustrating to all of us.

A related problem is the restriction we've traditionally placed on the length of programs. Again, this has to do with an unpleasant side of magazine publishing. Sadly, we've had to reject some excellent programs merely because they were too long to print. There's a limit to how much typing a reader is willing to undertake, even to get an exceptional program. Recently we've stretched this limit near the bursting point. We believe that programs like our *SpeedScript* word processor and *SpeedCalc* spreadsheet—with versions for Commodore, Atari, and Apple computers—are the best applications ever offered by a computer magazine. But both programs were written entirely in ML and required readers to spend many hours typing in thousands of numbers. Our MLX machine language entry utility is a partial solution. So is our COMPUTE! DISK. But we can't assume every reader is going to buy the disk, so we still have to restrict the length of programs to keep them accessible to all of our readers.

The new generation of high-powered, low-cost personal computers—exemplified by the Atari ST series—is allowing us to rethink our approach to program publishing. As the hardware grows more powerful, so does the software. The programs printed in magazines have to keep up, too. Some people go so far as to say that the days of program-oriented magazines are coming to an end. We strongly disagree. Consistently, reader feedback tells us that our programs and programming tutorials are the most popular features of our magazines. We feel that many useful programs can still be written in BASIC, and that BASIC will continue to be the language of choice for home programmers for some time to come. But to turn out really exceptional pieces of work, more and more programmers will be forced to turn to alternatives—particularly compilers. And their programs will grow larger and larger.

To meet this challenge, we're taking an exciting new approach. The details of this approach are part of the surprise we're preparing. For now, however, we can say this much:

We'll consider Atari ST program submissions written in practically any programming language you want. Have you written a utility in C for designing character fonts? Have you discovered a way to implement drop-down menus in ST BASIC? Have you written a general-purpose database manager in Prolog? Or an educational program in Pascal? Or a terminal program in Forth? Or an arcade-style game in machine language? Or a text editor in Modula-2?

Whatever it is, we'd like to see it. But don't get the idea that we're not picky. As always, we're interested in obtaining only the best-quality programs and articles we can find. If necessary, these programs can be much longer than ones we'd ordinarily publish in printed form. Of course, we still prefer to see programs which are as efficiently written as possible, so don't get carried away.

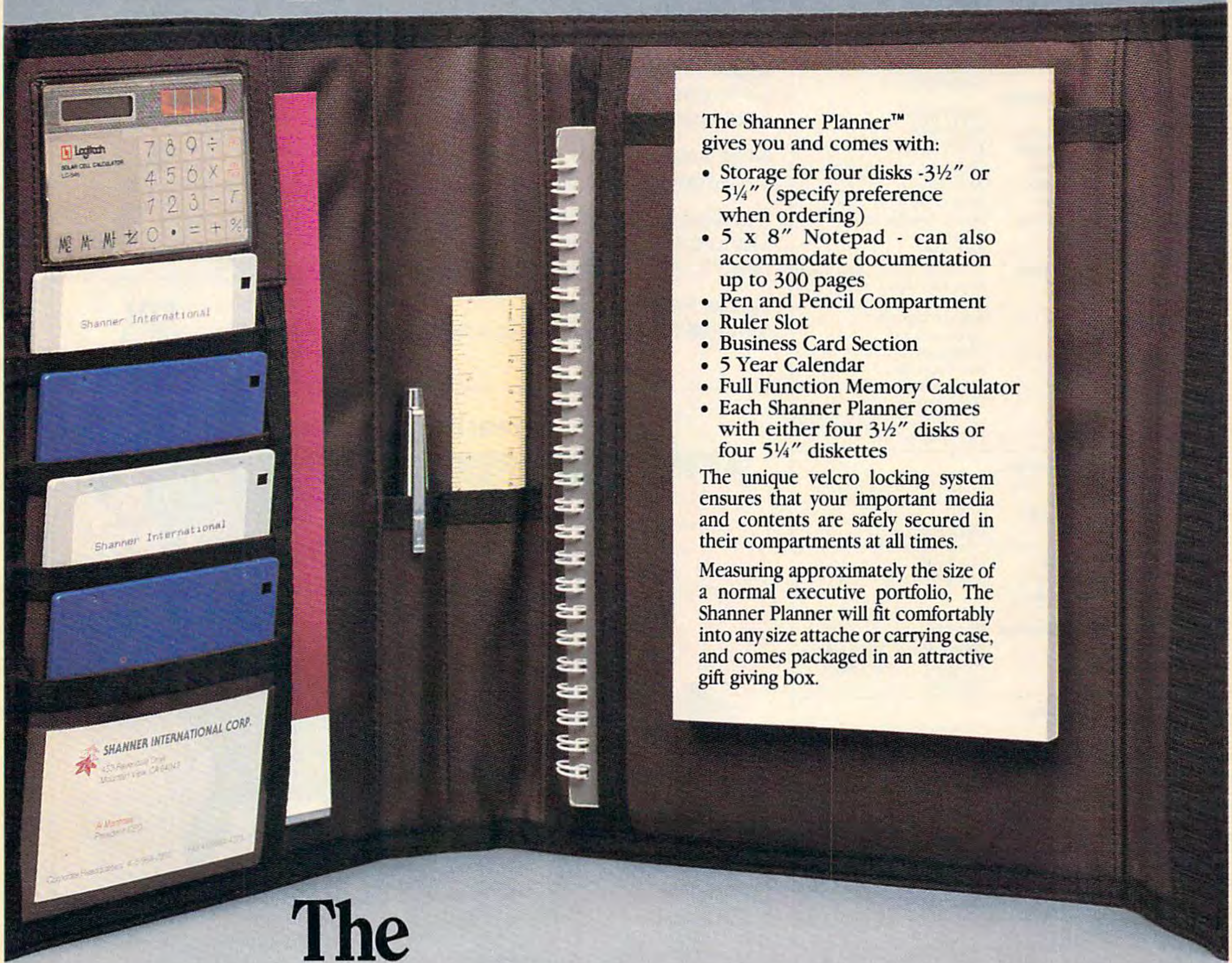
There's only one restriction: *The executable object code of the program must be legally usable by someone who doesn't own a copy of the language.* For instance, if you write a program with a compiled language, the compiled code must be a self-standing run-time package that anyone can load and run, whether or not they own the compiler. And we must be able to legally distribute the run-time package without becoming entangled in licensing fees and so forth. If you aren't sure about this, check with the company which produces the language.

Aside from this minor restriction, the gates are wide open. As a further incentive, we can hint that because of the way we'll be publishing these programs, some significant royalties may be in store for those whose work is accepted.

This is going to be an exciting experiment for all ST enthusiasts—readers, programmers, and those of us at COMPUTE!. Let's all make it a success.

Tom R. Halfhill, Editor

It speaks for itself.



The Shanner Planner™

$39.95

Dealer Inquiries Accepted
Call 800/828-6637

Remember the expression "A picture is worth a thousand words?" Take a second look. The Shanner Planner™ is the first ever full system portfolio created specifically for the computer user. Designed to address the individual needs of the computer industry, The Shanner Planner is tastefully constructed of long lasting, durable, textured nylon *(it's the in thing today)* to maintain its smart appearance over time.

SHANNER INTERNATIONAL CORP.

The Shanner Planner™ is a creation of, marketed worldwide by, and is a Trademark of SHANNER INTERNATIONAL CORP.

Please send me ________ Shanner Planner(s).

I am enclosing $ ________.
(CA residents add $2.60 tax per Planner)
Please make check or money order payable to:
Shanner International Corp.
453 Ravendale Drive
Mountain View, CA 94030

Canadian Orders:
Canadian cost per Planner is $49.95. (Ontario residents add $3.50 PST per Planner) Please direct orders to:
Shanner Sales & Marketing, Inc.
450 Garyray Drive
Weston, Ontario

Name ____________________

Address ____________________

City ________ State ________ Zip ________

For all orders: Please add $3.00 per Planner for shipping and handling. C.O.D.'s and purchase orders will not be accepted. Please allow 21 days for delivery.

Publisher	James A. Casella
Founder/Editor in Chief	Robert C. Lock
Senior Editor	Richard Mansfield
Managing Editor	Kathleen Martinek
Executive Editor	Selby Bateman

Editor	Tom R. Halfhill
Assistant Editor	Philip Nelson
Production Director	Tony Roberts
Production Editor	Gail Cowper
Editor, COMPUTE!'s GAZETTE	Lance Elko
Technical Editor	Ottis R. Cowper
Assistant Technical Editor	George Miller
Program Editor	Charles Brannon
Assistant Editor, COMPUTE!'s GAZETTE	Todd Heimarck
Assistant Features Editor	Kathy Yakal
Programming Supervisor	Patrick Parrish
Editorial Programmers	Tim Victor, Kevin Mykytyn
Research/Copy Editor	Joan Rouleau
Submissions Reviewer	Mark Tuttle
Programming Assistants	David Florance, David Hensley
Executive Assistant	Debi Nash
Administrative Assistants	Julia Fleming, Iris Brooks, Mary Hunt, Sybil Agee
Associate Editors	Jim Butterfield Toronto, Canada Harvey Herman Greensboro, NC Fred D'Ignazio Birmingham, AL David Thornburg Los Altos, CA
Contributing Editor	Bill Wilkinson

COMPUTE!'s Book Division Editor	Stephen Levy
Assistant Editors	Gregg Keizer, Ann Davies
Director, Book Sales & Marketing	Steve Voyatzis

Production Manager	Irma Swain
Art & Design Director	Janice R. Fary
Assistant Editor, Art & Design	Lee Noel
Mechanical Art Supervisor	De Potter
Artists	Debbie Bray, Dabney Ketrow
Typesetting	Terry Cash, Carole Dunton
Illustrator	Harry Blair

Director of Advertising Sales	Peter Johnsmeyer
Associate Advertising Director	Bernard J. Theobald, Jr.
Production Coordinator	Kathleen Hanlon

Promotion Assistant	Caroline Dark

Customer Service Manager	Diane Longo
Dealer Sales Supervisor	Orchid Tamayo
Individual Order Supervisor	Cassandra Green
Receptionist	Anita Armfield
Warehouse Manager	John Williams

Data Processing Manager	Leon Stokes

James A. Casella, President
Richard J. Marino, Vice President, Advertising Sales

COMPUTE! Publications, Inc. publishes:

COMPUTE! The Journal For Progressive Computing

COMPUTE!'s GAZETTE For VIC-20 And Commodore 64 Personal Computers

COMPUTE! Books

COMPUTE!'s GAZETTE DISK

COMPUTE!'s Apple Applications Special

Editorial offices:	324 West Wendover Avenue Suite 200 Greensboro, NC 27408 USA
Corporate offices:	825 7th Avenue New York, NY 10019 212-265-8360
Customer Service:	800-346-6767 (In NY 212-887-8525)
Hours:	9:30 A.M.–4:30 P.M. Monday–Friday

Coming In Future Issues

Hex War: Strategic Simulation For Commodore 64, 128, Atari 400/800/XL/XE, Atari ST And Others

The Top Five Public-Domain Programs For Your Computer

Super-Wide Printouts For Commodore: A Better Way To Print Spreadsheets And Wide Documents

Atari Sound Development System For Atari 400/800/XL/XE

The Screen Machine II Menu-Driven Drawing Program For IBM PC, PCjr, Compatibles

Apple Printer Master Get Full Power From Your Printer

Subscription Orders

COMPUTE!
P.O. Box 10954
Des Moines, IA 50340

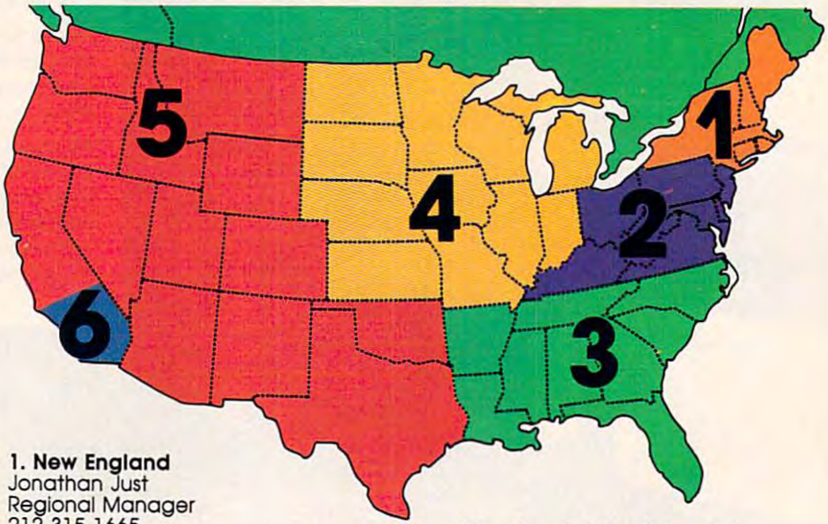
TOLL FREE Subscription Order Line
800-247-5470
In IA 800-532-1272

COMPUTE! Subscription Rates (12 Issue Year):

US	(one yr.)	$24
	(two yrs.)	$45
	(three yrs.)	$65
Canada and Foreign Surface Mail		$30
Foreign Air Delivery		$65

Member Audit Bureau of Circulations — Magazine Publishers Association

Advertising Sales



1. New England
Jonathan Just
Regional Manager
212-315-1665

2. Mid Atlantic
Jonathan Just
Regional Manager
212-315-1665

3. Southeast & Foreign
Harry Blair
919-275-9809

4. Midwest
Gordon Benson
312-362-1821

5. Northwest/ Mountain/Texas
Phoebe Thompson
Dani Nunes
408-354-5553

6. Southwest
Ed Winchell
213-378-8361

Director of Advertising Sales:
Peter Johnsmeyer

Associate Advertising Director:
Bernard J. Theobald, Jr.

COMPUTE! Home Office 212-887-8460.

Address all advertising materials to:
Kathleen Hanlon
Advertising Production Coordinator
COMPUTE! Magazine
324 West Wendover Avenue
Suite 200
Greensboro, NC 27408

The COMPUTE! subscriber list is made available to carefully screened organizations with a product or service which may be of interest to our readers. If you prefer not to receive such mailings, please send an exact copy of your subscription label to: COMPUTE! P.O. Box 10955, Des Moines, IA 50950. Include a note indicating your preference to receive only your subscription.

CLIP THIS PAGE TO FIND YOUR CLOSEST BOWATER AND BROWN DISC DEALERS CARRYING OUR NEW FREE SOFTWARE OFFER!

BOWATER COMPUTER FORMS INC.

CALIFORNIA
- TECHNICA
  ALBANY — 415/525-9404
- ALKO OFFICE SUPPLY
  BERKELEY — 415/848-3356
- SIERRA COMPUTER SUPPLIES
  CAMPBELL — 408/374-7810
- MICRO AGE COMPUTER STORE
  CONCORD — 415/680-1489
- WORLD OF COMPUTERS
  CONCORD — 415/825-1571
- WORLD OF COMPUTERS
  DUBLIN — 415/829-7211
- KINGS OFFICE PRODUCTS
  FAIRFIELD — 707/427-2933
- DEAN MARK OFFICE SUPPLY, INC.
  FREMONT — 415/791-0344
- DEAN MARK OFFICE SUPPLY, INC.
  FREMONT — 415/490-7448
- LEDEENS COMPUTER & OFFICE SUPPLY
  FREMONT — 415/793-3600
- DISCOUNT OFFICE SUPPLY
  MT. VIEW — 415/962-8900
- COMPUTER STORE OF OAKLAND
  OAKLAND — 415/763-7900
- WORLD OF COMPUTERS
  OAKLAND — 415/451-0212
- UNIVERSITY STATIONERS
  PALO ALTO — 415/326-7970
- DEAN MARK OFFICE SUPPLY, INC.
  PLEASANTON — 415/426-0282
- ALAMO COMPUTER
  SAN JOSE — 408/227-0891
- COMPUTER SUPPLIES UNLIMITED
  SAN JOSE — 408/265-0200
- DEAN MARK OFFICE SUPPLY, INC.
  SAN JOSE — 408/436-1833
- TELEMEDIA
  SAN JOSE — 408/435-7330
- M & M OFFICE SUPPLY
  SAN RAMON — 415/829-7283
- SANTA ROSA COMPUTER
  SANTA ROSA — 707/528-1291
- VORHEE'S COMPUTERS
  SANTA ROSA — 707/575-1419
- PALACE ART & OFFICE SUPPLY
  SOQUEL — 408/476-3799
- FRYS ELECTRONICS
  SUNNYVALE — 408/733-1770
- SUNNYVALE STATIONERS
  SUNNYVALE — 408/739-7000
- WORLD OF COMPUTERS
  WALNUT CREEK — 415/946-1070

COLORADO
- MAGNA-TRONICS WEST
  GOLDEN — 303/237-5558

FLORIDA
- ACS SERVICES INC.
  TALLAHASSEE — 904/681-6566

GEORGIA
- B & W ENTERPRISES
  WAYCROSS — 912/285-7286

INDIANA
- DIVERSIFIED DATA SYSTEMS
  INDIANAPOLIS — 317/297-5177

KANSAS
- WESTERN OFFICE SUPPLY
  COLBY — 913/462-3923
- COMPUTER GIN
  DODGE CITY — 316/225-2628
- THOROUGHBRED COMPUTERS
  EMPORIA — 316/342-0221
- BAAN TECH BUSINESS PRODUCTS INC.
  GARDEN CITY — 316/275-7419
- COMPUTER TEK OFFICE PRODUCTS
  GARDEN CITY — 316/275-7514
- COMPUTERLAND OF GREAT BEND
  GREAT BEND — 316/792-3524
- WESTERN OFFICE SUPPLY
  HAYES — 913/625-0027
- COMPUTERLAND OF HUTCHINSON
  HUTCHINSON — 316/662-6832
- THE COMPUTER GENERATION
  INDEPENDENCE — 316/331-8903
- COMPUTERARK
  LAWRENCE — 913/841-0094
- FORESIGHT SOLUTIONS dba COMPUTER OUTLET
  LAWRENCE — 913/842-7526
- CUSTOM BUSINESS SYSTEMS INC.
  MANHATTEN — 913/537-4306
- ADVANCE COMPUTER TECH
  McPHERSON — 316/241-5093
- THE COMPUTER ROOM
  McPHERSON — 316/241-8800
- COMPUTERLAND OF SALINA
  SALINA — 913/823-1555
- MICROAGE OF SALINA
  SALINA — 913/823-7596
- COMPUTRAIN INC.
  TOPEKA — 913/272-6800
- MEEKS INC.
  TOPEKA — 913/272-4750
- STAUFFER EQUIPMENT & SUPPLY CO.
  TOPEKA — 913/295-1157
- THOROUGHBRED COMPUTERS
  TOPEKA — 913/232-8089
- SOUTH CENTRAL BUSINESS FORMS INC.
  WICHITA — 316/264-0220
- WILBUR E. WALKER CO., INC.
  WICHITA — 316/267-2231
- ABEL SERVICES
  WINFIELD — 316/221-7577
- VARIETY HAUS
  ULYSSES — 316/356-3820

KENTUCKY
- NATIONAL OFFICE SUPPLY
  COVINGTON — 606/431-5100
- SUPERIOR PAPER COMPANY
  LOUISVILLE — 502/583-1647

MASSACHUSETTS
- COMPUTER DISCOUNT
  NORFOLK — 617/528-4186

MICHIGAN
- RETAIL COMPUTER CENTER, INC.
  BIRMINGHAM — 313/644-4820
- TIERRA
  CLARKSTON — 313/625-2511
- RETAIL COMPUTER CENTER, INC.
  FARMINGTON HILLS — 313/626-3240
- RETAIL COMPUTER CENTER, INC.
  GARDEN CITY — 313/422-2570
- OFFTECH
  GRAND RAPIDS — 616/451-8561
- MICRO WORLD COMPUTER CENTER
  LIVONIA — 313/427-0102
- MICRO WORLD COMPUTER CENTER
  MILFORD — 313/685-7766
- COMMUNIGRAPHIX, INC.
  ROSEVILLE — 313/445-8970
- MICRO WORLD COMPUTER CENTER
  ROSEVILLE — 313/771-3600
- MICRO WORLD COMPUTER CENTER
  SOUTHGATE — 313/285-0101
- LAKES INC.
  UNION LAKE — 313/360-0681

MINNESOTA
- BUDGET SOFTWARE
  CRYSTAL — 612/535-4369
- COMPUTER SATISFACTION
  ELK RIVER — 612/441-4225
- TEAM ELECTRONICS
  MANKATO — 507/387-7937
- PARAGON COMPUTERS
  ROCHESTER — 507/285-1999
- SOFTWARE CENTRE INTERNATIONAL
  ROSEVILLE — 612/631-3580

NEW JERSEY
- MDI COMPUTER STORE
  ENGLISHTOWN — 201/462-4600
- PERIPHERALS PLUS, INC.
  HOWELL — 201/363-6270
- STONEHENGE COMPUTER
  SUMMIT — 201/277-1020
- COMPUTER SERVICES CO.
  UPPER SADDLE RV — 201/327-8811
- COMPUTER OUTLET
  WESTWOOD — 201/666-8808

NEW MEXICO
- ANDATA
  LOS ALAMOS — 505/662-0031
  800/552-0070 in the mountain time zone

NEW YORK
- COMPUTER SOFT CENTER
  BREWSTER — 914/279-5001
- CORBIT BUSINESS MICRO SYSTEMS
  BRIGHTWATERS — 516/665-7811
- PRIME TIME COMPUTER
  BROOKLYN — 718/232-7770
- SYSUT COMPUTERS
  BROOKLYN — 718/743-8303
- VIDEO COMPUTER BIN
  BROOKLYN — 718/241-1993
- GA COMPUTER PRODUCTS
  BUFFALO — 716/854-0004
- MONROE CAMERA & COMPUTER
  MONROE — 914/782-8525
- COMPUTER OUTLET
  NEW CITY — 914/638-3800
- CIRO PRINTING & OFFICE SUPPLIES
  NEW YORK — 212/406-1323
- GERSTEN BUSINESS MACHINE
  NEW YORK — 212/764-6500
- HI TEC COMPUTERS
  NEW YORK — 212/374-1008
- INPUT COMPUTER
  NEW YORK — 212/725-9561
- SIMPLY COMPUTER & OFFICE PRODUCTS
  NEW YORK — 212/732-9336
- STATEWIDE OFFICE SUPPLY
  NEW YORK — 212/213-0770
- VENUS OFFICE SUPPLY CORP.
  NEW YORK — 212/477-2000
- COMPUTER CENTER
  ROCHESTER — 716/262-3166
- GA COMPUTER PRODUCTS
  ROCHESTER — 716/342-0602
- GA COMPUTER PRODUCTS
  SYRACUSE — 315/425-1414
- COMPURITE CENTRE
  WHITE PLAINS — 914/946-0900

NORTH CAROLINA
- MICRO BASICS
  CHARLOTTE — 704/527-1370

OHIO
- TETRA SYSTEMS
  AKRON — 216/666-2978
- MID-WEST MICRO
  ST. PARIS — 513/663-5488
- R.L. STEINMAN & SON
  ARLINGTON — 419/365-5555
- SOFTWARE CENTRE
  CANTON — 216/492-9163
- COPY SOURCE
  CINCINNATI — 513/489-8821
- IMAGING SUPPLIES
  CINCINNATI — 513/871-2525
- MIDWEST COMPUTER SUPPLIES
  CINCINNATI — 513/481-0020
- SCHEAR COMPUTER SUPPLIES
  CINCINNATI — 513/729-5522
- DISKOUNT SOFTWARE
  COLUMBUS — 614/231-9910
- BEST LITTLE WAREHOUSE
  CUYAHOGA FALLS — 216/920-1136
- CAD CAM
  DAYTON — 513/293-3381
- EXPRESS BUSINESS FORMS
  DAYTON — 513/866-1788
- PC NETWORK SYSTEMS
  DAYTON — 513/435-2256
- TETRA SYSTEMS
  DAYTON — 513/299-3476
- COMPUTER WAREHOUSE
  EUCLID — 216/481-5515
- SOFTWARE CONNECTION
  MAYFIELD HEIGHTS — 216/473-2722
- VALCOM
  MAYFIELD HEIGHTS — 216/449-4030
- FREDERICK COMPUTER PRODUCTS
  MIDDLETOWN — 513/424-3233
- COMPUTER SHOWCASE
  NILES — 216/652-2571
- SOFTWAIRE CENTER
  NORTH OLMSTED — 216/734-4755
- BIG BYTE COMPUTERS
  POLAND — 216/758-0009
- COMPUTER DISCOUNTS
  REYNOLDSBURG — 614/863-6701

PENNSYLVANIA
- WEAN ASSOCIATES
  CONSHOHOCKEN — 215/834-0866
- SUBURBAN OFFICE SUPPLIERS
  LANSDALE — 215/362-6510

SOUTH CAROLINA
- S.T.O.P. INC.
  CAYCE — 803/796-5600
- MICRO COMPUTER DEPOT
  SUMTER — 803/775-5166

TENNESSEE
- ECLECTIC SOLUTIONS CORPORATION
  CLEVELAND — 615/336-3658

TEXAS
- SHRIVER OFFICE SUPPLY
  ALICE — 512/664-2638
- DICK OFFICE SUPPLY
  AUSTIN — 512/258-7968
- INFORMATION PROCESSING SUPPLY
  AUSTIN — 512/445-7223
- KELLY OFFICE PRODUCTS
  CEDAR PARK — 512/258-1656
- A & D OFFICE SUPPLY
  CORPUS CHRISTI — 512/883-1865
- EL PASO MICRO MART INC.
  EL PASO — 915/595-1188
- STEELMAN OFFICE PRODUCTS
  GONZALES — 512/672-9535
- DICK OFFICE SUPPLY
  HARLINGEN — 512/428-1221
- COMPUTERS, INC.
  HOUSTON — 713/556-1331
- EXCEL BUSINESS SYSTEMS, INC.
  HOUSTON — 713/952-0236
- COOK OFFICE MACHINES
  McALLEN — 512/686-9576
- DICK OFFICE SUPPLY
  McALLEN — 512/682-6306
- COMAL OFFICE PRODUCTS
  NEW BRAUNFELS — 512/625-9108
- PERMIAN MICRO MART
  ODESSA — 915/367-6179
- ALL AMERICAN OFFICE SUPPLY
  PLANO — 214/424-8524
- WILDCAT COMPUTING, INC.
  PLANO — 214/424-3583
- HOUSE OF SUPPLIES
  RAYMONDVILLE — 512/689-2524
- AB COMPUTER ELECTRONICS CENTER
  ROUND ROCK — 512/244-2090
- THE COMPUTER STORE
  SAN ANGELO — 915/942-7505
- AMERICAN OFFICE PRODUCTS
  SAN ANTONIO — 512/654-8808
- AZTEC BUSINESS FORMS
  SAN ANTONIO — 512/654-7969
- BUSINESS WORLD
  SAN ANTONIO — 512/820-0103
- PAUL ANDERSON CO.
  SAN ANTONIO — 512/734-8111
- PREMIER BUSINESS PRODUCTS
  SAN ANTONIO — 512/680-4149
- PROFESSIONAL BUSINESS PRODUCTS
  SAN ANTONIO — 512/822-5544
- TA OFFICE SUPPLY
  SAN ANTONIO — 512/377-3500
- WAGNER BROTHERS COMPUTER STORE
  SAN ANTONIO — 512/657-2012
- COMMERCIAL OFFICE PRODUCTS
  SAN MARCOS — 512/353-7477
- KLIPPLE BUSINESS PRODUCTS
  SAN MARCOS — 512/396-5855
- STEELMAN OFFICE PRODUCTS
  SEGUIN — 512/372-3950
- VICTORIA OFFICE EQUIPMENT
  VICTORIA — 512/573-4371

BROWN DISC MANUFACTURING, INC.

TO LOCATE **SCHOLAR** DEALERS IN YOUR AREA CALL:

NATIONWIDE
- ALVIN/MODERN SCHOOL SUPPLY
  1-800-234-2329
  CT: 203/243-8991
- RICHARD YOUNG OFFICE PRODUCTS
  1-800-325-0136
  FL: 305/979-3100
- NASCORP INC.
  1-800-321-3883
  OH: 216/774-1831

NORTHWEST
- ECZEL CORP.
  1-800-431-2489
  WA: 206/872-7636

ROCKY MOUNTAIN
- COMPUTER SERVICES BROKER
  1-950-1088 at dial tone 770-185
  CO: 303/697-0561

ALABAMA
- DUCHESS COMPUTER PRODUCTS
  GRAND BAY — 205/865-6324
- INTERSTATE SCHOOL SUPPLY & EQUIPMENT COMPANY OF ARKANSAS, TENNESSEE, ALABAMA
  1-800-222-8600
  MS: 601/948-8600

ARKANSAS
- INTERSTATE SCHOOL SUPPLY & EQUIPMENT COMPANY OF ARKANSAS, TENNESSEE, ALABAMA
  1-800-222-8600
  MS: 601/948-8600

COLORADO
- COLBORN'S
  DENVER — 303/778-1220

FLORIDA
- BRANDS MART
  N. MIAMI — 305/624-5400
- COLLEGE MANAGEMENT SERVICE, INC.
  FL: 305/783-3100
- DUCHESS COMPUTER PRODUCTS
  AL: 205/865-6324
- INTERSTATE SCHOOL SUPPLIERS
  1-800-241-4336
  GA: 404/997-6714

GEORGIA
- DUCHESS COMPUTER PRODUCTS
  AL: 205/865-6324
- INTERSTATE SCHOOL SUPPLIERS
  1-800-241-4336
  ATLANTA — 404/997-6714

ILLINOIS
- DUCHESS COMPUTER PRODUCTS
  AL: 205/865-6324
- MIDWEST COMPUTER SUPPLY
  WEST CHICAGO — 312/231-1112

INDIANA
- COLLEGE MANAGEMENT SERVICE, INC.
  FL: 305/783-3100
- MIDWEST COMPUTER SUPPLY
  IL: 312/231-1112

KENTUCKY
- MENTOR SYSTEMS
  LEXINGTON — 606/231-0519

LOUISIANA
- INTERSTATE COMPANIES OF LOUISIANA
  1-800-272-9800
  BATON ROUGE — 504/387-5131
  NEW ORLEANS — 504/581-9500

MARYLAND
- DUCHESS COMPUTER PRODUCTS
  AL: 205/865-6324

MISSISSIPPI
- DUCHESS COMPUTER PRODUCTS
  AL: 205/865-6324
- MISSISSIPPI SCHOOL SUPPLY COMPANY OF MISSISSIPPI
  1-800-222-8600
  JACKSON — 601/948-8600

MONTANA
- COLBORN'S
  BILLINGS — 406/245-3158

NEBRASKA
- COLLEGE MANAGEMENT SERVICE, INC.
  FL: 305/783-3100

NEW JERSEY
- COLLEGE MANAGEMENT SERVICE, INC.
  FL: 305/783-3100

NEW MEXICO
- COLBORN'S
  ALBUQUERQUE — 505/292-3225
- SPARTAN II ELECTRONICS, INC.
  ALBUQUERQUE — 505/345-3479

NORTH DAKOTA
- COLBORN'S
  GRAND FORKS — 701/746-5441

SOUTH CAROLINA
- INTERSTATE SCHOOL SUPPLIERS
  1-800-241-4336
  GA: 404/997-6714

TENNESSEE
- DUCHESS COMPUTER PRODUCTS
  AL: 205/865-6324
- INTERSTATE SCHOOL SUPPLY & EQUIPMENT COMPANY OF ARKANSAS, TENNESSEE, ALABAMA
  1-800-222-8600
  MS: 601/948-8600

TEXAS
- SPARTAN II ELECTRONICS, INC.
  NM: 505/345-3479

WYOMING
- COLBORN'S
  CASPER — 307/265-5026

Ask about other BROWN DISC AND BOWATER SPECIAL OFFERS. BROWN DISC ALSO GIVES YOU 100 SHEET PACKAGES OF FREE BOWATER COMPUTER PAPER WITH THE PURCHASE OF A SCHOLAR 2 PACK.

AND,

BOWATER GIVES YOU 1 FREE BROWN DISKETTE WITH 500 SHEETS AND 2 FREE BROWN DISKETTES WITH 2500 SHEETS OF BOWATER COMPUTER PAPER.

So, see our dealers today to receive these value packed products!

Readers Feedback

The Editors and Readers of COMPUTE!

If you have any questions, comments, or suggestions you would like to see addressed in this column, write to "Readers' Feedback," COMPUTE!, P.O. Box 5406, Greensboro, NC 27403. Due to the volume of mail we receive, we regret that we cannot provide personal answers to technical questions.

SpeedScript's Lineage

What are the differences between *SpeedScript 3.0, 3.1, 3.2,* and so on?

Leo Mitchener

Here's the genealogy of SpeedScript *for the Commodore 64: The original 64* SpeedScript *(now called version 1.0) appeared in the January 1984 issue of* COMPUTE!'s GAZETTE. *A slightly modified version (1.1) appeared in* COMPUTE!'s Second Book of Commodore 64. *The next major update,* SpeedScript 2.0, *appeared only on the premier* GAZETTE DISK *in May 1984. Like the original, its title screen did not include a version number; however, it can be distinguished from other versions by its custom character set and help screen.*

Version 3.0 made its debut in the March 1985 issue of COMPUTE! *and on the special* COMPUTE! DISK *for that month. It can easily be distinguished from its predecessors because the command line says* SpeedScript 3.0. *Corrections for several minor bugs were published in the May 1985 "CAPUTE!" column. With these enhancements, the title on the screen indicates version 3.1. It was this version which appeared in the book* SpeedScript: The Word Processor for the Commodore 64 and VIC-20, *and on the companion disk for that book. Further corrections—most notably a fix for an underlining bug—appeared in the article* "SpeedScript 3.0 *Revisited" in the December 1985 issue of* COMPUTE!; *these enhancements changed the version number on the screen to 3.2. Version 3.2 also appeared on the January 1986* COMPUTE! DISK. *The corrections in the December article included the changes from version 3.1, so it is possible to upgrade directly from 3.0 to 3.2.*

As in many other areas of personal computing, there isn't any official rule that dictates how program versions are to be numbered. For SpeedScript *we've followed what seems to be the most common convention. In general, a whole number difference (such as 2.0 versus 3.0) signals a major enhancement, while a fractional change (3.0 versus 3.2) indicates minor enhancements. Unless otherwise indicated, a reference to one member of a group is also applicable to the others. We usually use* SpeedScript 3.0 *to refer to all members of the version 3 family: 3.0, 3.1, and 3.2. For example, the 3.0 version of the POKEs given in the January "Readers' Feedback" to make* SpeedScript *default to disk or tape also works for 3.2, even though this was not stated explicitly.*

For a description of how SpeedScript *3.0 differs from previous versions in terms of features, see the article in the March 1985 issue of* COMPUTE!.

The VIC-20 version of SpeedScript *3.0 appeared in the April 1985 issue of* COMPUTE!. *The Atari and Apple versions of* SpeedScript *start with version 3.0 and made their debut in the May 1985 and June 1985 issues of* COMPUTE!, *respectively.*

Machine Language Delays

I have recently written a program in 6502 machine language for the VIC-20. I want to have a one- or two-second pause between the title screen and the main program, but I don't know how to make one.

Stephen Brown

One way to create a delay in machine language (ML) is to use a do-nothing loop much as you would in BASIC. For instance, the BASIC loop shown here pauses for about one second on a VIC:

```
FOR TD=1 TO 1000:NEXT
```

A similar machine language loop looks like this:

```
        LDY #0
WAIT    DEY
        BNE WAIT
        RTS
```

This loop creates a delay, but only for a fraction of a second. To produce a longer delay, you could use two nested loops:

```
        LDY #0
        LDX #0
WAIT    DEY
        BNE WAIT
        DEX
        BNE WAIT
        RTS
```

This loop delays for about a second. For longer delays you can use more nested loops combining different memory locations and registers. Some computers have a built-in clock that's available for the same purpose. On the Commodore 64 and VIC-20, for instance, location 162 is incremented every 1/60 second by the computer's hardware interrupt routine. To create a delay with the built-in clock, store a zero in location 162, then wait until it reaches the number of seconds you want to delay divided by 60. This short routine creates a three-second delay:

```
        LDA #0
        STA 162
WAIT    LDA 162
        CMP #180
        BNE WAIT
        RTS
```

Changing Apple Proofreader's Checksum

I am using an Apple IIe with a color TV as a monitor. One problem with the TV is that reverse characters are difficult if not impossible to read. Is there any way to modify the "Apple Automatic Proofreader" so the checksum numbers appear normal instead of reverse? I am not the best typist in the world and was delighted to find a Proofreader program. But the checksum numbers are so hard to read that I can't use it at all.

Robert A. Love

Attention Programmers

COMPUTE! magazine is currently looking for quality articles on Commodore, Atari, Apple, and IBM computers (including the Commodore Amiga and Atari ST). If you have an interesting home application, educational program, programming utility, or game, submit it to COMPUTE!, P.O. Box 5406, Greensboro, NC 27403. Or write for a copy of our "Writer's Guidelines."

It's easy to defeat the reverse video effect for Apple computers. Run the Proofreader as usual, then enter this line in direct mode (without a line number):

POKE 804,176: POKE 806,186: POKE 822,176: POKE 824,186

The checksum numbers appear in the usual screen location in normal video. Since this modification makes the checksum harder to distinguish from other numbers on the screen, you probably won't want to make this change unless it's absolutely necessary.

Scrolling Atari Messages

I am an Atari 1200XL owner. I would like to know how to move a message like 1=LOAD 2=LOCK 3=UNLOCK across the screen.

Bobby Chan

The following BASIC program scrolls any message up to 100 characters across the top of the screen. The variable MESS$ in line 20 contains the message to be printed. You can reposition the scrolling message to any line on the screen by changing the POSITION statement in line 30.

```
KK 5 DIM MESS$(100),TEMP$(20
     0):L=1
EC 10 PRINT "{CLEAR}":POKE 8
      2,0:POKE 752,1
JO 20 MESS$="TYPE LETTER TO
      RUN, OR 1=LOAD 2=LOCK
      3=UNLOCK 4=EXIT... ":N
      =LEN(MESS$)
FE 25 TEMP$(1,N)=MESS$:TEMP$
      (N+1,2*N)=MESS$
DB 30 POSITION 0,0:PRINT TEM
      P$(L,L+39)
AM 40 L=L+1:IF L>2*N-40 THEN
      L=1
EB 50 FOR TD=1 TO 50:NEXT TD
      :GOTO 30
```

IBM BASIC Directory

Can you tell me how to read and display the disk directory on an IBM PC from within a BASIC program?

Kamal Ashour

There are two simple ways to approach this. The first is simply to print the directory to the screen at the appropriate time in your BASIC program. A second method would be to read the directory into a string array for use by your program at some later point. Here's a short routine that employs the first method:

```
EG 100 REM FSPEC$="A:*.*":GOTO 1
       40
MD 110 PRINT:PRINT"Select drive:
        (";:COLOR 16,15:PRINT"A
       B";:COLOR 7,0:PRINT CHR$(
       29)CHR$(29)"/"CHR$(28)")"
HB 120 DRIVE$=INKEY$+":":A=ASC(D
       RIVE$):IF (A OR 32)<97 OR
        (A OR 32)>98 THEN 120
IF 130 DRIVE$=CHR$(A AND 223)+":
       ":FSPEC$=DRIVE$+"*.*"
GG 140 ON ERROR GOTO 150:FILES F
       SPEC$:ON ERROR GOTO 0:END
JL 150 BEEP:COLOR 31:CLS:PRINT "
       Cannot read directory":CO
       LOR 7:ON ERROR GOTO 0:END
```

This routine will ask you from which drive (A: or B:) you want to read the directory. If you have a single-drive system (drive A: only), remove the REM from line 100. Here's another routine that uses the second method:

```
KJ 1000 REM FSPEC$="A:*.*":GOTO
        1040
LH 1010 PRINT:PRINT"Select drive
        : (";:COLOR 16,15:PRINT"
        A B";:COLOR 7,0:PRINT CH
        R$(29)CHR$(29)"/"CHR$(28
        )")"
HC 1020 DRIVE$=INKEY$+":":A=ASC(
        DRIVE$):IF (A OR 32)<97
        OR (A OR 32)>98 THEN 102
        0
ME 1030 DRIVE$=CHR$(A AND 223)+"
        :":FSPEC$=DRIVE$+"*.*"
LN 1040 DEF SEG=0:WIDTH 80
IF 1050 HEAD=1050:TAIL=1052:BUFF
        ER=1054
OJ 1060 CLS:COLOR 23,0,0:PRINT"R
        eading disk directory"
OE 1070 COLOR 0:ON ERROR GOTO 10
        90
EJ 1080 FILES FSPEC$:ON ERROR GO
        TO 0:GOTO 1100
IL 1090 BEEP:COLOR 31:CLS:PRINT
        "Cannot read directory":
        COLOR 7:ON ERROR GOTO 0:
        END
HF 1100 DIM TT$(24):LOCATE 3,1:C
        OLOR 7:ROWS=0
CD 1110 POKE HEAD,30:POKE TAIL,3
        4:POKE BUFFER,0:POKE BUF
        FER+1,79:POKE BUFFER+2,1
        3:POKE BUFFER+3,28:'Put
        code for End, Enter into
         keyboard
DE 1120 LINE INPUT TT$(ROWS):IF
        TT$(ROWS)<>"" THEN ROWS=
        ROWS+1:GOTO 1110
DN 1130 IF NOT DIMMED THEN DIM F
        $(ROWS*4-1):DIMMED=1
JP 1140 ROWS=ROWS-1:FOR I=0 TO R
        OWS:FOR J=0 TO 3
KE 1150 T$=MID$(TT$(I),J*18+1,12
        )
KD 1160 IF T$<>"" THEN F$(ENTRIE
        S)=T$:ENTRIES=ENTRIES+1
KA 1170 NEXT J:NEXT I:ERASE TT$:
        ENTRIES=ENTRIES-1:DEF SE
        G:RETURN
```

This routine reads the filenames from the disk directory into an array named F$. One advantage of this method is that you need to look only once at the directory. Once the directory information is stored in a string, you can extract the filenames whenever it's convenient and print them in any format you like. With a little more programming, you could cursor through the directory to access a particular file, sort the directory entries alphabetically, catalog all your disks, or whatever. Again, remove the REM from line 1000 if you have a single-drive system.

64 RAM Report

Can you give me a short program that tests the RAM in my 64? I have had trouble running a particular BASIC program and think that my computer must have a defective RAM chip.

Fred Wayne

Though it's tempting to blame the hardware when things go awry, RAM chips rarely fail. Every time you turn on a Commodore 64, it performs a RAM verification as part of its normal power-up sequence. It tests every RAM address from location 1024 (the start of screen memory) upward until it hits a ROM (Read Only Memory) location that can't be written to. Unless a cartridge is installed, the test includes all of the BASIC programming space (locations 2048–40959).

Here's how the power-up test works. After saving the original contents of the tested memory location, the computer stores the value 85 ($55) there, then reads the contents back to make sure the operation was successful. Then it stores the value 170 ($AA) there and reads the contents again. To understand why those particular values are used, look at them in binary form:

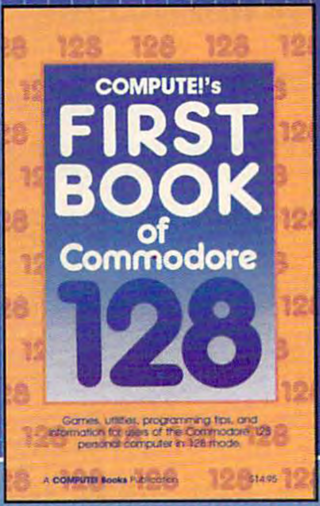
01010101 = $55 = 85
10101010 = $AA = 170

As you can see, every one bit in the first number is a zero bit in the second and vice versa. While you could test a location by successively writing and reading back every value from 0 to 255 (the maximum range for a single address), this method checks whether you can write and read back a one and a zero in each of the location's eight bits—which amounts to much the same thing. If a RAM address passes both tests, the 64 restores its original contents and proceeds to the next higher location, stopping as soon as it finds a read-back value that doesn't match what was just written. This normally happens at location 40960, the start of BASIC ROM. The location just below that (40959) is used as the top of BASIC memory.

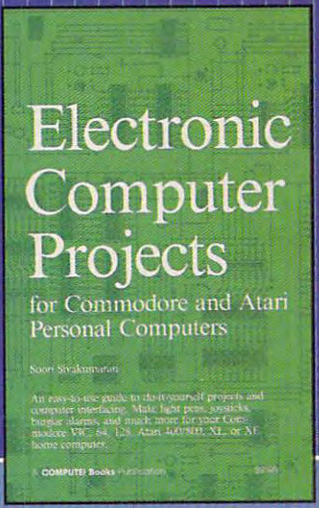
Later in the startup sequence, the 64 subtracts 2048 from the top-of-memory value to calculate the number of bytes free for the startup message. Since 40959 − 2048 = 38911, the familiar message 38911 BASIC BYTES FREE tells you that the 64 just wrote and read back two values for every address in BASIC program space without detecting any errors.

If you're not convinced by the built-in test, here's a short ML program that tests the 64's RAM somewhat more thoroughly, writing and reading back every value from 0 to 255 before it concludes that a RAM address is functional. Be sure to save the program before you run it since

Two Exciting New Books



from COMPUTE!



COMPUTE!'s First Book of the Commodore 128

A spectacular collection of articles and programs exclusively for the Commodore 128 *in 128 mode.*

Edited

The editors at COMPUTE! Publications have collected some of the best games, programs, and tutorials for the Commodore 128 from *COMPUTE!* and *COMPUTE!'s Gazette*, plus some never-before-published articles and programs. Learn how to create windows, program sound, and make disks autoload. You'll even find a map of all the important memory locations. There's something for every 128 user. All programs run in 128 mode. *A disk is available which includes programs in the book, $12.95.*

$14.95 ISBN 0-87455-059-9

Electronic Computer Projects

Learn how to build all kinds of new devices to interface with your computer from inexpensive, available parts.

For the Commodore 64, 128, VIC, and any eight-bit Atari personal computer.

Soori Sivakumaran

This introduction to digital electronics and computer interfacing is the easy way to learn how computers interact with the outside world. Using a Commodore 64, 128, VIC, or any eight-bit Atari computer and *Electronic Computer Projects*, you'll be guided through the steps to building a joystick, light pen, game paddle, and numerous other devices. And since each project is independent from the others, you can choose only those projects that interest you. All the projects can be built at home and most require fewer than half a dozen parts.

$9.95 ISBN 0-87455-052-1

Visit your local book or computer store for these new titles. Or order directly from COMPUTE! Books. Call toll-free 800-346-6767 (in NY 212-887-8525) or write COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Please include $2.00 per book ($5.00 air mail) for shipping and handling. NC residents add 4.5 percent sales tax. Allow 4–6 weeks from receipt of order for delivery.

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

COMPUTE! books are available in the U.K., Europe, the Middle East, and Africa from Holt Saunders, Ltd., 1 St. Anne's Road, Eastbourne, East Sussex BN21 3UN, England, and in Canada from McGraw-Hill, Ryerson Ltd., 330 Progress Ave., Scarborough, Ontario, Canada M1P 2Z5.

the ML portion erases the BASIC loader:

```
FK 10 ADR=49152
JG 20 READ BYT:IF BYT<>256 THE
      N POKE ADR,BYT:ADR=ADR+1
      :CK=CK+BYT:GOTO 20
RC 30 IF CK<>11516 THEN PRINT"
      ERROR IN DATA STATEMENTS
      --CHECK TYPING":END
JA 40 PRINT "PRESS RETURN TO C
      HECK BASIC RAM":PRINT
FB 50 PRINT "SYS 49152"CHR$(14
      5)CHR$(145)CHR$(145)
DR 49152 DATA 169,0,133,251,16
         9,8,133,252,32,228,25
         5
PD 49158 DATA 208,58,166,251,1
         65,252,32,205,189,169
         ,32
SQ 49164 DATA 32,210,255,160,0
         ,162,0,202,138,145,25
         1
FP 49170 DATA 209,251,240,18,1
         52,72,138,72,169,72,1
         60
JA 49176 DATA 192,32,30,171,10
         4,170,104,168,76,59,1
         92
DQ 49182 DATA 224,0,208,226,23
         0,251,208,2,230,252,1
         65
RQ 49188 DATA 252,201,160,208,
         193,96,157,95,18,66,6
         5
BX 49194 DATA 68,146,32,0,256
```

*This program checks the 51199 RAM locations from 2048 to 53247, which includes all of BASIC program space as well as the 8K of RAM underneath BASIC ROM and the 4K RAM zone from 49152 to 53247. If a location passes the test, its address is printed. If not, you'll see the message BAD in reverse video with an arrow pointing to the address. Since it performs over 13 million (51199*256) read/write operations, this program takes about 15 minutes to run. You can cut it short by pressing any key.*

Format With PRINT USING

I am having difficulty formatting an amortization table on my PCjr that will display dollars to two decimal places (to the cents place). Currently, my program drops the trailing zeros following a decimal point. Do you have a solution for this?

Keith Bovee

The answer is to substitute PRINT USING for the more common PRINT statement. PRINT USING is very versatile and can be used to format the output of string or numeric data. The general format for this command is:

PRINT USING *format$; expression(s)*

Replace format$ *with a string constant or variable containing special formatting characters (listed in your BASIC manual). The formatting characters tell the computer exactly how it should print* the expression that follows the semicolon. *The expression may be either string or* *numeric data, and you may include more than one expression.*

Perhaps the most common use of PRINT USING is to format numeric data, a task that requires only two formatting characters. The number sign (#) reserves a digit position within the output string, and the dollar sign ($) stands for a dollar sign. For instance, type the following lines in direct mode (without line numbers):

```
X = 1234.00
PRINT X
PRINT USING "$####.##";X
```

The first PRINT statement strips the decimal places, printing 1234. That's normal in BASIC, but undesirable in a program that requires dollars and cents format. The PRINT USING statement prints $1234.00 complete with a dollar sign and two decimal places. You can find additional examples of how to use PRINT USING in the IBM BASIC manual.

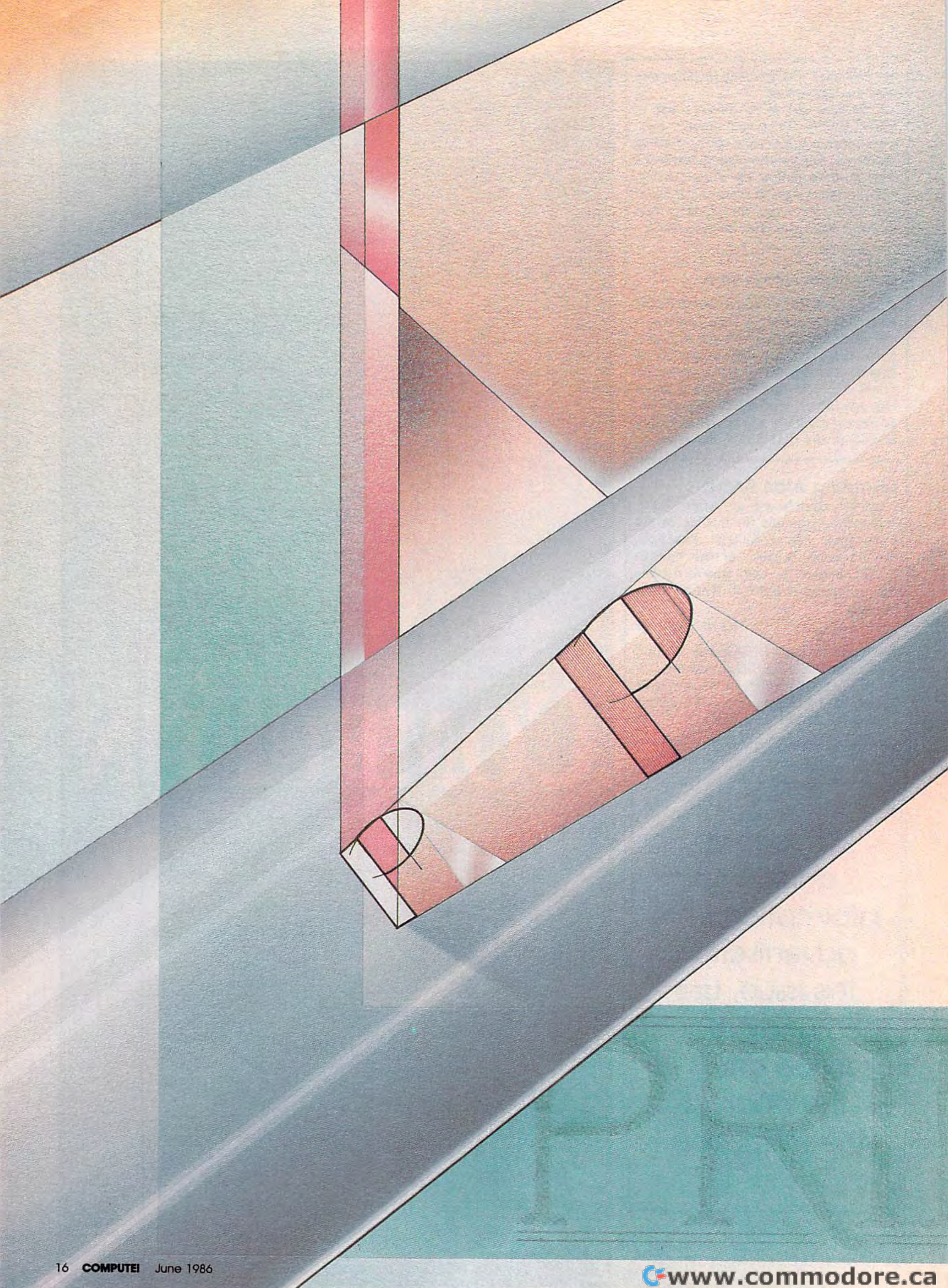
Disabling Atari BASIC

I recently purchased an Atari 800XL and some programs for it. One of the programs, *Micro League Baseball*, doesn't work because of the built-in BASIC. Is there an easy way to disable the computer's built-in BASIC temporarily?

Chris Greatens

To disable the built-in BASIC on an Atari 600XL, 800XL, or XE, hold down the OPTION key when turning on the computer. On an Atari 400, 800, or 1200XL, simply unplug the BASIC cartridge. ©

To receive additional information from advertisers in this issue, use the handy reader service cards in the back of the magazine.



The Changing Face Of Printer Technology

Joan Rouleau, Research/Copy Editor

Today's printers are better than ever: They're faster, quieter, more versatile, less expensive, and produce higher-quality output than even their recent predecessors. Innovative new technologies—such as lasers and LED arrays—are offering more choices for home computer owners, while the more established technologies—such as dot-matrix, daisywheel, and ink-jet printers—have been greatly refined. Here's a look at some of the changes that are reshaping the printer marketplace.

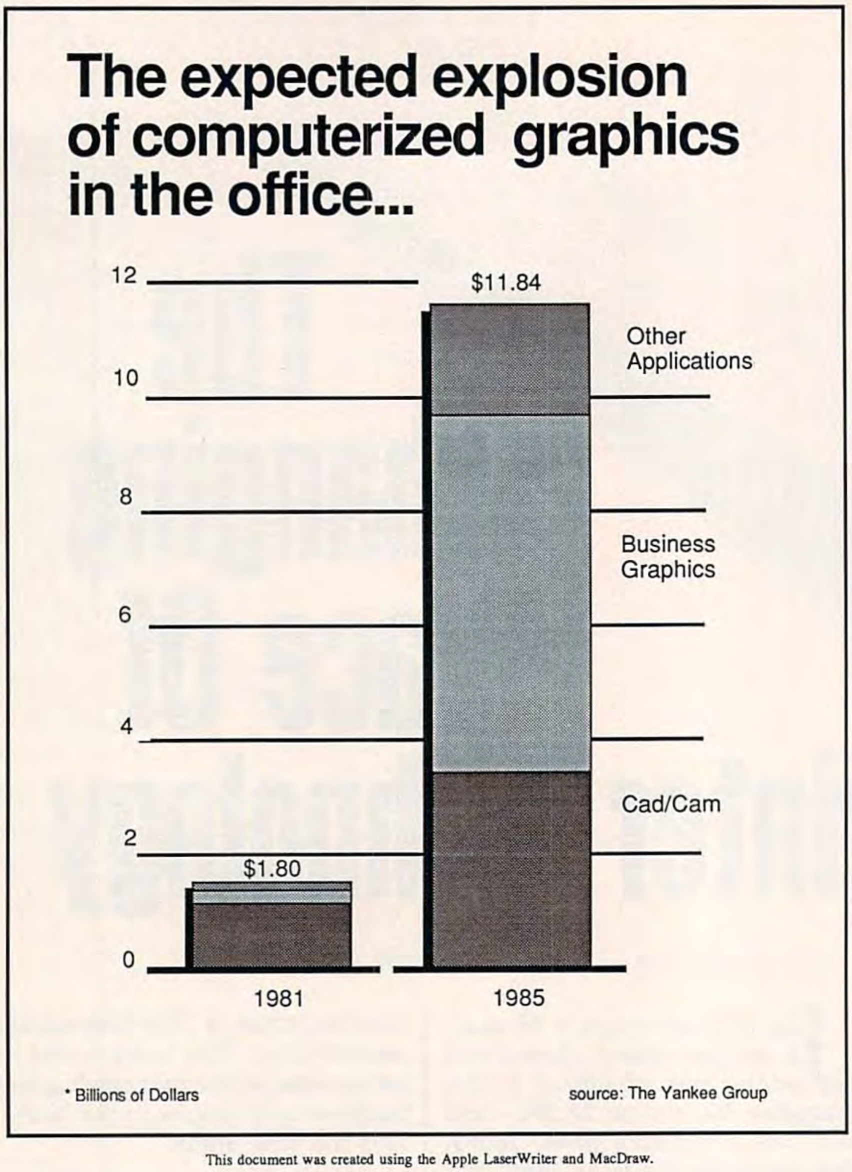
Just five years ago, a 40-characters-per-second daisywheel printer was advertised in this magazine for almost $2,000—and that was a discount price. Today, that same cash can buy a silent, six-page-per-minute, multiple-font laser printer. Similarly, it wasn't very long ago that the blocky, awkward type produced by dot-matrix printers was appropriate only for printing draft copy. Now, with print resolution as great as 300 dots per inch in some new models, dot-matrix printers are reaching true letter quality. Better yet, the intense competition among manufacturers and retailers continues to push prices down and spawn a wider selection of features.

Printers are becoming such an integral part of the home computerist's workroom that only in a technical sense can they still be considered peripherals. The percentage of home computer owners with a printer nearly doubled between 1983 and 1985: from 28 percent to 53 percent, according to Link Resources, a New York market research firm. This trend is tied to an increase in word processing and business applications in the home, says the Link study.

Not only are more people buying printers, but they're also expecting more from the printers they buy. In particular, more and more people want better-quality print. A recent survey by another market research firm, Frost & Sullivan of New York, named print clarity as the single most important factor among those choosing a letter-quality printer. Other factors were ease of repair, long life, and then price.

Answering this demand for better print is a wide array of new nonimpact printers and substantial improvements in dot-matrix printers. Laser printers, once affordable only by large businesses, have just begun to drop in price. Within a couple of years, they, too, may become a contender in the home printer arena.

Noel



A sample of near-typeset quality output from a laser printer.

Laser printers were originally developed about a decade ago for use with mainframe computers, and they work much like photocopy machines. In a photocopier, the original is illuminated with a bright light that transfers the image of the page onto a light-sensitive drum. Through a thermal and electrochemical process, the drum then fuses the image onto another sheet of paper.

A laser printer works in a similar way, but uses a low-power laser to scribe the images onto the drum. Therefore, it is a *page printer*—it prints a whole page at a time rather than a single character at a time, like most printers. The newest laser printers can print up to a fleeting 12 pages per minute. And unlike most dot-matrix or daisywheel printers, they run quietly.

In the last couple of years, improvements in laser and photocopy technology brought the price of laser printers down to the $7,000 range, making them accessible to considerably more businesses. Then, last fall, QMS of Mobile, Alabama, introduced its Kiss laser printer for only $1,995, bringing this technology within reach of small businesses and some home users.

Among the other manufacturers who are developing laser printers in the $2,000 to $3,000 range are Okidata, Canon, Mannesmann Tally, Dataproducts, and ITT Qume. Many industry watchers predict that a $1,000 laser printer will be available by the end of 1987.

Others, however, are more skeptical about how soon the laser printer will become a mainstay in the home. Laser printers are still quite costly to manufacture, they argue, and it may be some time before these costs go down. Virtually all of the mechanisms for laser printers are made in Japan, and the devaluation of the dollar against the yen may keep laser printers more expensive for a while.

Perhaps in light of these considerations, some manufacturers are looking to other nonimpact technologies for their page printers. Particularly favored among several manufacturers is the light-emitting diode (LED) array. LEDs are tiny semiconductors that emit light when energized by a pulse of current, often seen as power indicator lights on stereos and computers. LED printers work something like laser printers, except they use an LED array to print the page instead of a laser. LED array printers are comparable in speed to laser printers, and because they have fewer moving parts, they are cheaper to produce and transport. Among the manufacturers who have chosen LED technology for their page printers are IBM and Datasouth.

Another nonimpact technology, *ion deposition*, also is making its debut. Instead of using light to transfer the image onto a drum, these printers shoot ion beams onto an electrically conductive drum.

$29.95.
A SMALL PRICE TO PAY TO FIRE UP YOUR INVESTMENTS.

Invest it in the Dow Jones News/Retrieval® Membership Kit, and sample the nation's premier online financial information resource. With five free hours* to explore, a User's Guide to point the way, and a subscription to *Dowline*—the magazine of News/Retrieval, filled with case studies and insights—the Membership Kit will show you how to find fuel for your investment ideas.

Monitor current stock quotes, or track historical trends for companies or industries that interest you. Call up earnings and economic forecasts, and reports from major brokerage houses. Get breaking news as early as 90 seconds after it appears on the newswires, and news from *The Wall Street Journal* (online exclusively with News/Retrieval) back to January 1, 1984—to see what impact government or world events are having on the marketplace.

You can access News/Retrieval with almost any computer and modem, terminal or communicating word processor.

The Dow Jones News/Retrieval Membership Kit. Use it to kindle *your* success. Without getting burned.

To order or for more information, call **1-800-345-8500, Ext. 337.** (Alaska, Hawaii and foreign, call 1-215-789-7008, Ext. 337.) Or use the coupon today.

YES! I want my investments fired up!
Send me the Dow Jones News/Retrieval® Membership Kit.
☐ **Personal offer - $29.95 Each** includes: • 5 Free Hours* • Single Password • User's Guide • 1 Year of *Dowline* • $12 Annual service fee waived for one year
☐ **Corporate offer** (for multiple users) - **$49.95 Each** includes: • 8 Free Hours* • Multiple Passwords • User's Guide • 1 Year of *Dowline* • $12 Annual service fee waived for one year
☐ **Send more information**

$________ ENCLOSED (Check or Money Order)
☐ Bill me later Charge to: ☐ AmExp ☐ MC ☐ VISA

Card No. ______________________ Exp. Date________

Signature ______________________

Name ______________________

Address ______________________

City/State/Zip ______________________

Daytime Phone Number ______________________

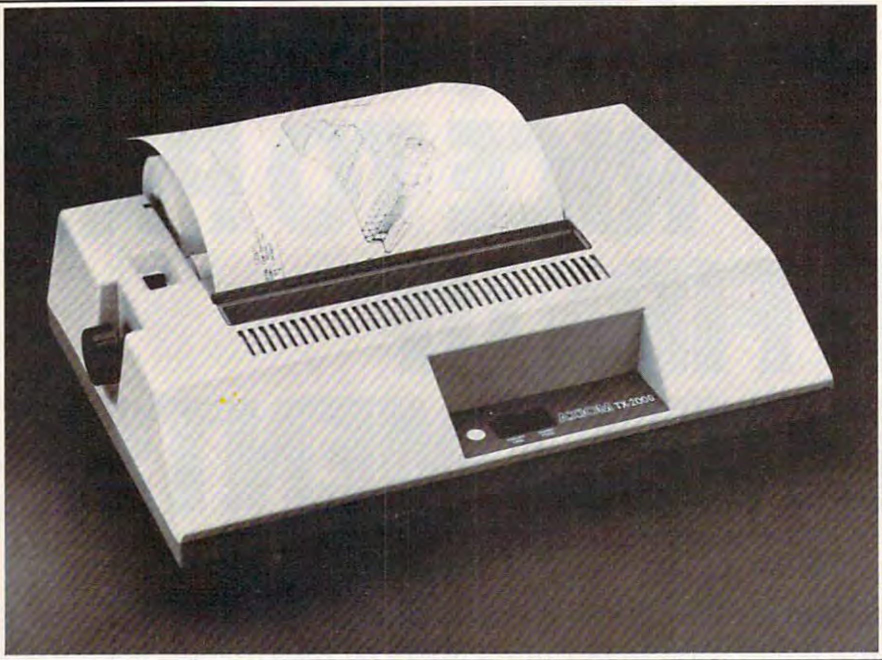
Computer Make & Model ______________________

Mail to: DJN/R Membership Kit, P.O. Box 300, Princeton, NJ 08543-0300
ATTN: John McGovern 5CO0506E

Dow Jones News/Retrieval®
Fuel for your ideas.

$29.95 Membership Kit also available at computer retailers and selected bookstores.

*Per account. Limited to new subscribers only. Free time (5 hours per Personal offer; 8 hours per Corporate offer) must be used within 30 days after receipt of password(s). Certain data bases have fees over and above usage charges which are excluded from free time offer.



Axiom's TX 2000 can print out a hardcopy of any video display.

Unlike laser or LED printers, ion deposition printers do not use heat as part of the transfer process. Companies investigating this technology include Star Micronics, Mannesmann Tally, and C. Itoh. Mannesmann's director of marketing, John Roberts, predicts that ion deposition printers are "the nonimpact technology that will probably come closest to replacing impact printers."

On another front, ink-jet printers continue to evolve and generate interest. These printers, as the name implies, spray a jet of ink from several tiny nozzles onto the paper. When first introduced, they could only produce draft-quality copy which had a tendency to smudge. Now major manufacturers such as Canon and Diablo are perfecting this technology and are reportedly developing high-resolution ink-jet printers.

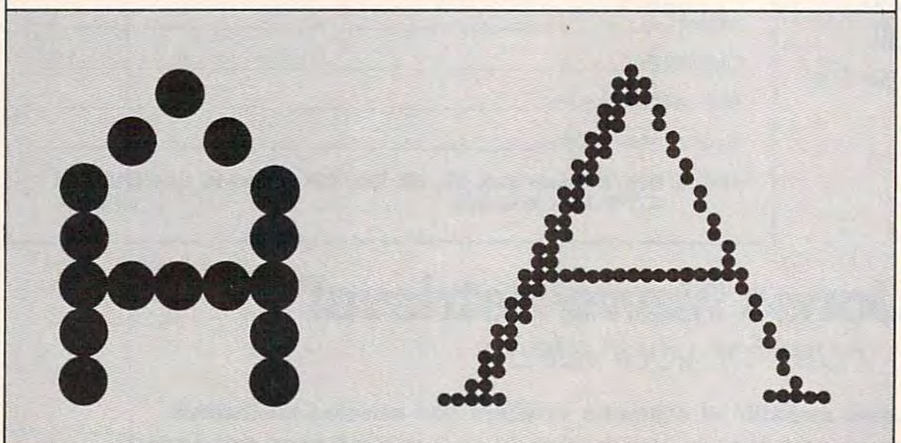
Nonimpact technology is still in its infancy and will likely undergo a great many changes—in speed, price, and sophistication—over the next few years. "Lasers have opened the door. We're finding that there are other doors," says Tom Bongiorno, director of marketing for Star Micronics. "Just as when the first dot-matrix printer came out, it was certainly a breakthrough. Then print quality became better, prices dropped to one third or less than initially...the quality continues to pick up and prices will probably still drop."

Does this surge of new nonimpact printers aimed at the home market mean the demise of dot-matrix? Not anytime soon. Dot-matrix printers are still considerably cheaper and have improved quite a lot over the past couple of years. Says Dennis Cox of Epson America, "There's continued optimism and growth in the dot-matrix industry. We're seeing more products become available, improved features, and new price levels."

Just in the last year, the resolution of dot-matrix print has greatly improved. All dot-matrix printers use a printhead which consists of a vertical row of stacked wires. As the printhead moves across the page, these wires are hammered onto the paper in different patterns to form characters in a rectangular matrix. When dot-matrix printers were first introduced, characters were formed in a 5 × 7 or 8 × 8 matrix (see the accompanying figure). Now several printers are on the market which have 18 or even 24 wires in their printheads. This allows the printer to form characters which are much better defined, and produce better graphics as well. Improvements have also been made which enable better positioning of the printhead, so even nine-wire printheads can produce higher-quality print than ever before.

Among the new high-resolution dot-matrix printers are Toshiba's P321 ($699), which features a 24-wire printhead, 80-column carriage, and multiple type font

A 5 × 7 character from a 8-pin printhead. *A 24 × 30 character from a 24-pin printhead.*



C-128™ and C-64™ REQUIRED READING

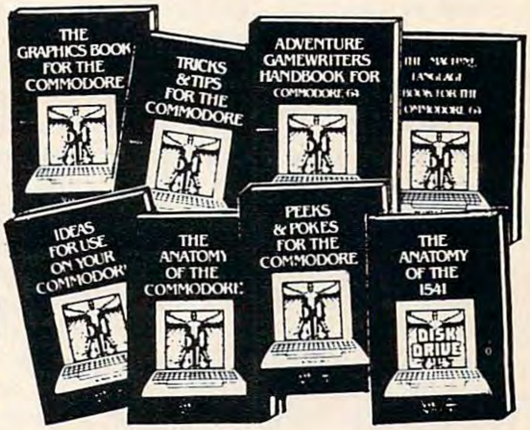
Detailed guide presents the 128's operating system, explains graphic chips, Memory Management Unit, 80 column graphics and commented ROM listings. **500pp** $19.95

Get all the inside information on BASIC 7.0. This exhaustive handbook is complete with commented BASIC 7.0 ROM listings. Coming Summer '86. $19.95

Filled with info for everyone. Covers 80 column hi-res graphics, windowing, memory layout, Kernal routines, sprites, software protection, autostarting. **300pp** $19.95

Insiders' guide for novice & advanced users. Covers sequential & relative files, & direct access commands. Describes DOS routines. Commented listings. $19.95

Learn fundamentals of CAD while developing your own system. Design objects on your screen to dump to a printer. Includes listings for '64 with Simon's Basic. **300pp** $19.95



Introduction to programing; problem analysis; thorough description of all BASIC commands with hundreds of examples; monitor commands; utilities; much more. $16.95

Presents dozens of programming quick-hitters. Easy and useful techniques on the operating system, stacks, zero-page, pointers, the BASIC interpreter and more. $16.95

Essential guide for everyone interested in CP/M on the 128. Simple explanation of the operating system, memory usage, CP/M utility programs, submit files & more. $19.95

ANATOMY OF C-64 Insider's guide to the '64 internals. Graphics, sound, I/O, kernal, memory maps, more. Complete commented ROM listings. **300pp** $19.95

ANATOMY OF 1541 DRIVE Best handbook on floppy explains all. Many examples and utilities. Fully commented 1541 ROM listings. **500pp** $19.95

REVISED & EXPANDED

MACHINE LANGUAGE C-64 Learn 6510 code write fast programs. Many samples and listings for complete assembler, monitor, & simulator. **200pp** $14.95

GRAPHICS BOOK C-64 - best reference covers basic and advanced graphics. Sprites, animation, Hires, Multicolor, lightpen, 3D-graphics, IRQ, CAD, projections, curves, more. **350pp** $19.95

TRICKS & TIPS FOR C-64 Collection of easy-to-use techniques: advanced graphics, improved data input, enhanced BASIC, CP/M, more. **275pp** $19.95

1541 REPAIR & MAINTENANCE Handbook describes the disk drive hardware. Includes schematics and techniques to keep 1541 running. **200pp** $19.95

ADVANCED MACHINE LANGUAGE Not covered elsewhere: - video controller, interrupts, timers, clocks, I/O, real time, extended BASIC, more. **210pp** $14.95

PRINTER BOOK C-64/VIC-20 Understand Commodore, Epson-compatible printers and 1520 plotter. Packed: utilities; graphics dump; 3D-plot; commented MPS801 ROM listings, more. **330pp** $19.95

SCIENCE/ENGINEERING ON C-64 In depth intro to computers in science. Topics: chemistry, physics, biology, astronomy, electronics, others. **350pp** $19.95

CASSETTE BOOK C-64/VIC-20 Comprehensive guide; many sample programs. High speed operating system fast file loading and saving. **225pp** $14.95

IDEAS FOR USE ON C-64 Themes: auto expenses, calculator, recipe file, stock lists, diet planner, window advertising, others. Includes listings. **200pp** $12.95

COMPILER BOOK C-64/C-128 All you need to know about compilers: how they work; designing and writing your own; generating machine code. With working example compiler. **300pp** $19.95

Adventure Gamewriter's Handbook Step-by-step guide to designing and writing your own adventure games. With automated adventure game generator. **200pp** $14.95

PEEKS & POKES FOR THE C-64 Includes in-depth explanations of PEEK, POKE, USR, and other BASIC commands. Learn the "inside" tricks to get the most out of your '64. **200pp** $14.95

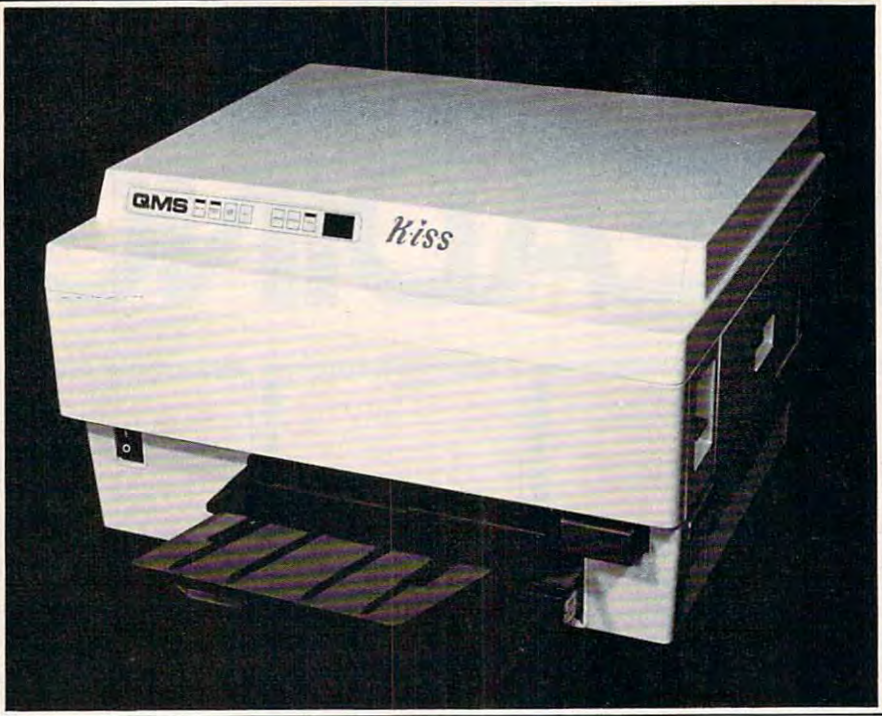
Optional Diskettes for books
For your convenience, the programs contained in each of our books are available on diskette to save you time entering them from your keyboard. Specify name of book when ordering. **$14.95 each**

C-128 and C-64 are trademarks of Commodore Business Machines Inc.

Abacus Software

P.O. Box 7219 Grand Rapids, MI 49510 - Telex 709-101 - Phone (616) 241-5510

Call **now** for the name of your nearest dealer. Or to order directly by credit card, MC, AMEX of VISA call **(616) 241-5510**. Other software and books are available–Call and ask for your **free** catalog. Add $4.00 for shipping per order. Foreign orders add $10.00 per book. Dealer inquires welcome–1400+ nationwide.



The first laser printer for under $2,000: The new Kiss from QMS.

cartridges. Okidata's 2410 ($2,395) also has a 24-wire printhead and can print 136 columns across. Star Micronic's NB-15 ($1,449), another 24-wire printer, produces letter quality at 100 characters per second (cps) and draft quality at 300 cps, and has a 16K buffer. NEC has a new line of dot-matrix printers which includes the 24-wire Pinwriter P5 ($1440–$1560). Fujitsu America's 24-wire dot-matrix printer has a liquid crystal display which shows the print status. Recently released 18-wire dot matrix printers include Mannesmann Tally's MT490 and Datasouth's DS440 (prices haven't been released for either machine). While these may still be a little too expensive for many home users, the prognosis for less costly high-resolution dot-matrix printers over the next few years is very good.

Dot-matrix printers are also improving in speed and other features. "What we feel is the trend for the dot-matrix market right now is that more and more features are being built into printers," says Frank Rexach, product manager for C. Itoh. Some of the features in C. Itoh's new C310 are 300 cps draft printing, paper feeding from the top, bottom, and rear, and all control keys located on the front panel.

The dot-matrix printer's chief rival, the daisywheel, stands a chance of being superseded by the letter-quality dot-matrix printers and the new nonimpact devices. Daisywheel printers work much like typewriters and used to be the only way to get letter-quality print. Now many manufacturers have slowed or stopped their production of daisywheels while expanding into the dot-matrix and nonimpact areas.

As Mannesmann Tally's John Roberts says, "Daisywheel manufacturers are the most subject to displacement by the laser printer." Or, as another manufacturer puts it, "I wouldn't want to be *only* in the daisywheel market right now."

Anyone who s ever tried to use a printer for graphics knows how difficult it can be. While virtually all dot-matrix printers have some graphics capability, there are no standard control codes for accessing this feature. Programs that print graphics have a hard time supporting all the different printers that are available.

This situation has led to the development of *page description languages.* With a page description language, your software can access features like graphics and text in several fonts without knowing what kind of printer you have. All that's necessary is that your printer understand the page description language that your software is generating. One of the most popular of these is *PostScript* by Adobe Systems, which can be used with the Apple LaserWriter and other printers.

Thanks to *PostScript,* high-quality printing is available to those who can't justify the expense of a laser printer for occasional printing jobs. A document description can be sent over a phone line with a modem, and some professional typesetting machines understand *PostScript.* So in some areas, it's already possible to create and lay out a document, upload it to a print shop that has one of these typesetters, and have it typeset without leaving your home.

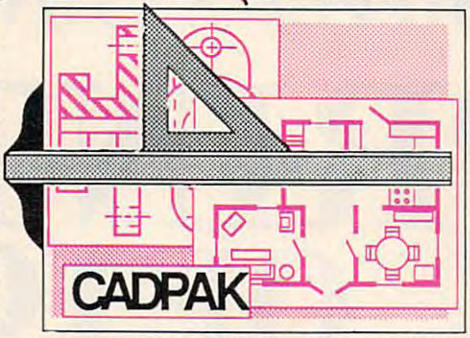
Other new developments in the area of printer graphics include the *digital videoprinter*—a printer which makes hardcopy from any type of raster-scan video display (including computer monitors and TVs). The TX 2000 videoprinter, recently released by Axiom Corporation, can capture a moving video image and even rotate or reverse the image. The TX 2000 lists for $1,995.

A number of new color printers are also opening up new graphics possibilities for home computers. Juki has just released a nine-wire color dot-matrix printer which can produce up to seven colors from a four-color ribbon. Fujitsu America is offering a color version of its 24-wire dot-matrix printer. And a few companies are developing color laser printers for use with personal computers. It'll be a while before these devices are found in many homes, though—color laser printers, like color photocopiers, are still very expensive.

Advances in printer technology continue at a rapid pace, and the printers that have been recently announced demonstrate just how quickly the market is progressing. Whether you need a simple and inexpensive dot-matrix printer for casual use or a state-of-the-art laser printer for near-typeset quality documents, the latest printers provide unprecedented performance at far lower prices. ©

'128™ and C-64™ SPECTACULAR SOFTWARE

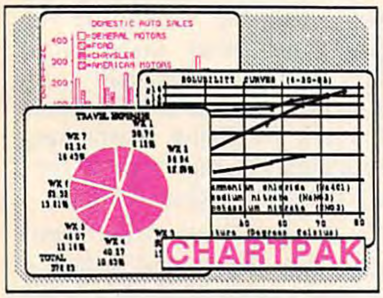
The *complete* compiler and development package. Speed up your programs 5x to 35x. Many options: flexible memory management; choice of compiling to machine code, compact p-code or both. '128 version: 40 or 80 column monitor output and FAST-mode operation. '128 Compiler's extensive 80-page programmer's guide covers compiler directives and options, two levels of optimization, memory usage, I/O handling, 80 column hi-res graphics, faster, higher precision math functions, speed and space saving tips, more. A great package that no software library should be without. **128 Compiler $59.95**
64 Compiler $39.95



Remarkably easy-to-use interactive drawing package for accurate graphic designs. New dimensioning features to create exact scaled output to all major dot-matrix printers. Enhanced version allows you to input via keyboard or high quality lightpen. Two graphic screens for COPYing from one to the other. DRAW, LINE, BOX, CIRCLE, ARC, ELLIPSE available. FILL objects with preselected PATTERNS; add TEXT; SAVE and RECALL designs to/from disk. Define your own library of symbols/objects with the easy-to-use OBJECT MANAGEMENT SYSTEM–store up to 104 separate objects. **C-128 $59.95**
C-64 $39.95

For school or software development. Learn C on your Commodore with our in-depth tutorial. Compile C programs into *fast* machine language. C-128 version has added features: Unix™-like operating system; 60K RAM disk for fast editing and compiling Linker combines up to 10 modules; Combine M/L and C using CALL; 51K available for object code; Fast loading (8 sec. 1571, 18 sec. 1541); Two standard I/O librarys plus two additional libraries—math functions (sin, cos, sqrt, etc.) & 20+ graphic commands (line, fill, dot, etc.). **C-128 $79.95**
C-64 $79.95

Not just a compiler, but a complete system for developing applications in Pascal with graphics and sound features. Extensive editor with search, replace, auto, renumber, etc. Standard J & W compiler that generates fast machine code. If you want to learn Pascal or to develop software using the best tools available–**SUPER Pascal** is your first choice. **C-128 $59.95**
C-64 $59.95



Easily create professional high quality charts and graphs without programming. You can immediately change the scaling, labeling, axis, bar- filling, etc. to suit your needs. Accepts data from CalcResult and MultiPlan. C-128 version has 3X the resolution of the '64 version. Outputs to most printers. **C-128 $39.95**
C-64 $39.95

PowerPlan

One of the most powerful spreadsheets with integraded graphics. Includes menu or keyword selections, online help screens, field protection, windowing,trig functions and more. PowerGraph, the graphics package, is included to create integrated graphs & charts. **C-64 $39.95**

COBOL Compiler for the C-64	$39.95
Ada Compiler for the C-64	$39.95
VideoBasic Language for the C-64	$39.95

OTHER TITLES AVAILABLE:

Technical Analysis System

Sophisticated charting and technical analysis system for serious investors. Charting and analyzing past history of a stock, TAS can help pinpoint trends & patterns and predict a stock's future. Enter data from the keyboard or from online financial services. **C-64 $59.95**

Personal Portfolio Manager

Complete protfolio management system for the individual or professional investor. Easily manage your portfolios, obtain up-to-the-minute quotes and news, and perform selected analysis. Enter quotes manually or automatically through Warner Computer Systems. **C-64 $39.95**

Xper

XPER is the first "*expert system*" for the C-128 and C-64. While ordinary data base systems are good for reproducing facts, XPER can derive knowledge from a mountain of facts and help you make expert decisions. Large capacity. Complete with editing and reporting. **C-64 $59.95**

Abacus Software

P.O. Box 7219 Grand Rapids, MI 49510 - Telex 709-101 - Phone (616) 241-5510

Call **now** for the name of your nearest dealer. Or to order directly by credit card, MC, AMEX of VISA call **(616) 241-5510**. Other software and books are available–Call and ask for your **free** catalog. Add $4.00 for shipping per order. Foreign orders add $12.00 per item. Dealer inquires welcome–1400+ nationwide.

Buyer's Guide To The PRINTERS OF 1986

If you already know how you'll be using a printer and what features you'll need before you start shopping, the hard part is over. There are many good printers available for a variety of applications, and prices continue to drop as manufacturers expand their hardware lines.

We've gathered information on printers in the under-$800 price range and listed some of the most important features in the following chart. Any omissions are not an editorial judgment of quality.

Here's a brief explanation of the major categories on the chart:

Compatibility. Chances are your computer has either a serial or parallel port (or both) that hooks up to a printer. Some printers come in either serial or parallel versions; some offer both interfaces; and some are available in parallel or serial only. If the printer you want comes only in a version that doesn't support your computer, you should be able to buy a separate interface that allows that configuration. Also, many printer manufacturers sell interfaces designed specifically for certain computers, avoiding any compatibility problems.

Be careful here. In some situations, a particular interface will let you print text, but will be incapable of producing graphics. If there's any doubt, it's best to try and test your setup at a computer dealer.

Print technology. This refers to how characters and graphics are actually transferred from printer to paper. There are three types in this price range: impact, thermal, and ink-jet.

Impact printers form characters by striking the paper through an inked ribbon, either with a *daisywheel* (a small wheel whose spokes have letters and numbers on their tips), or with a printhead containing a column of tiny wires or pins that form characters and graphics *(dot-matrix). Thermal* printers use either a column of hot pads that change the color of heat-sensitive paper, or a column of tiny spark plugs that evaporate a special aluminum coating onto the paper, exposing an underlying dark surface. Thermal printers require special paper, which often costs more than regular paper and has a shorter life. *Thermal transfer* printers work with any kind of paper because they use ribbons; heat from the printhead melts a waxlike ink onto the paper. *Ink-jet* printers spray ink onto the paper through tiny holes.

Speed. How fast does the printer operate? This can vary if the printer offers different modes. *Draft mode* is usually the fastest, but produces rougher, fainter type. *Near letter quality (NLQ),* or *correspondence mode,* takes longer to print, but looks more polished. Some printer speeds vary depending on the type of *font* (for example, pica or elite) used. In our chart, a wide speed range, like 30–120 characters per second (cps), indicates that the printer offers some kind of correspondence-quality type.

Pitch. This indicates how many characters fit on a line, measured in characters per inch (cpi) or characters per line (cpl). The pitch range for a printer often varies greatly, especially if it is capable of printing several types of fonts.

Buffer. A buffer is an area of memory in a printer that can store a fixed amount of text while the printer is working, freeing up the computer for other tasks. Most printers in the under-$800 price range still have rather small buffers, so if you'll be doing many long printing jobs, you may want to consider buying an add-on buffer.

Feed type. *Friction-feed* printers grip the paper and move it around the platen much as a typewriter does, while *tractor-feed* printers have teeth at both ends of the platen that grab holes at the edges of continuous-feed paper. Many printers have optional tractors.

Suggested retail price. This is the price set by the manufacturer; you may well find it at a lower price if you shop around.

A full explanation of the graphics capabilities of each printer takes more space than we have available. If you plan to use your printer extensively for printing graphics, make sure it's capable of doing what you need before you buy.

Model Name	Manufacturer/ Distributor	Compatibility	Print Technology	Speed	Pitch	Buffer	Feed Type	Warranty	Suggested Retail Price	Comments
Alphacom Aero	Alphacom, Inc.	Parallel or serial std	Dot matrix	130 cps	5-16.5 cpi	2K	Friction and pin std	6 months	299	Dot addressable and fully programmable graphics
Alphapro 101	Alphacom, Inc.	Parallel or serial std	Daisy wheel	20 cps	10–15 cpi	93 characters	Friction std	6 months	399.95	
Image Writer	Apple Computer	Serial std	Dot matrix	120 cps	4.5–17 cpi	32K available	Friction or tractor std	90 days	495	
Image Writer II	Apple Computer	Compatible with Apple II, III, Lisa	Dot matrix	180–250 cps	4.5–17 cpi	32K available	Friction and adjustable width pin std; sheet feeder opt	90 days	595	
Scribe	Apple Computer	Serial std	Dot matrix	50–80 cps	10–17 cpi	N/A	Tractor std	90 days	249	
Aprotek Daisy 1120	Aprotek	Parallel std; serial opt	Daisy wheel	20 cps	10–15 cpi and proportional	2K	Friction std; tractor and cut sheet feeder opt	1 year	299.95	Two week trial available
Elite 5	Aprotek	Parallel and Commodore serial std	Daisy wheel	10–12 cps	10 cpi	None	Friction std; tractor opt	1 year	174.95	Two week trial available;Commodore version $184.95
SP-1000	Aprotek	Parallel or direct connect; IBM standard	Dot matrix	20–70 cps	10-15 cpi	1.5K	Friction and tractor std	2 years	239.95	Dot addressable graphics; Commodore graphics built-in on Commodore version ($219.95)
DX 1500	Axiom Corp.	Parallel std	Daisy wheel	14 cps	10–12 cpi and proportional	2K	Friction std; tractor and auto cut sheet feeder opt	1 year	349	
DX 2000	Axiom Corp.	Parallel std	Daisy wheel	20 cps	10–15 cpi and proportional	2K	Friction std; tractor opt	1 year	449	
DX 2500	Axiom Corp.	Parallel std; serial opt	Daisy wheel	20 cps	10–15 cpi and proportional	2K	Friction std; tractor opt	1 year	499	
DX 3500	Axiom Corporation	Parallel and serial std	Daisy wheel	35 cps	10–15 cpi and proportional	2K	Friction std; tractor opt	1 year	699	
Thin Print 80P/80S	Axonix Corp.	Parallel or serial std	Thermal transfer dot matrix	40 cps	10–17 cpi	2K	Friction std	90 days	339	High resolution graphics; portable (battery powered), AC adaptor included;
Thin Print 100	Axonix Corp.	Parallel or serial std	Thermal transfer dot matrix	25-100 cps	10-17 cpi	2K	Friction std	90 days	299	
Thin Write 80W	Axonix Corp.	Parallel and serial std; HP-1L opt	Dot matrix	24–100 cps	5–17 cpi	2K	Friction std; tractor opt	90 days	449	Battery powered
Thin Write 100	Axonix Corp.	Parallel and serial std	Dot matrix	25-100 cps	5-17 cpi	4K	Friction and pin std; tractor opt	90 days	479	
Blue Chip 120/NLQ	Blue Chip Electronics	Parallel std	Dot matrix	120 cps	5–17 cpi	3 lines	Tractor std	6 months	279	
D12/10	Blue Chip Electronics	Commodore serial std	Daisy wheel	12 cps	10 cpi	2K	Friction std; tractor opt	6 months	249	Comes with Fleetwriter III wordprocessor
D20/10	Blue Chip Electronics	Parallel and Commodore serial std	Daisy wheel	20 cps	10 cpi	2K	Friction std; tractor opt	6 months	279	Comes with Fleetwriter III wordprocessor
M 120/10	Blue Chip Electronics	Parallel std; serial opt	Dot matrix	120 cps	5–17 cpi	3 lines (4K opt)	Friction and tractor std	6 months	229	Dot addressable graphics
M 150/15	Blue Chip Electronics	Parallel std; serial opt	Dot matrix	130 cps	5–17 cpi	2K	Friction and tractor std	6 months	349	Dot addressable graphics
HR-10	Brother International Corp.	Parallel and serial std	Daisy wheel	12 cps	10–15 cpi and proportional	2K	Friction and tractor std	90 days	349	
M-1109	Brother International Corp.	Parallel and serial std	Dot matrix	25-100 cps	10 cpi	2K	Friction std; tractor opt	1 year	269	
M-1509	Brother International Corp.	Parallel and serial std	Dot matrix	45–180 cps	10 cpi	3K	Friction and tractor std; cut sheet feeder opt	1 year	549	Seven bit image graphics
Prowriter Jr.	C. Itoh	Parallel std; serial opt	Dot matrix	20–105 cps	10–17 cpi	1 line	Friction and tractor std	1 year	329	
Prowriter 8510 S series	C. Itoh	Serial or parallel std	Dot matrix	45–180 cps	10–17 cpi	2K	Tractor and friction std	1 year	499	
Prowriter C-310	C. Itoh	Parallel std; serial opt	Dot matrix	28–300 cps	10–15 cpi	2K	Friction and tractor std	1 year	599	
Y10-20	C. Itoh	Serial or parallel std	Daisy wheel	22 cps	10–15 cpi	2K	Friction std; tractor opt	1 year	549	
Legend 808	CAL-ABCO	Parallel std	Dot matrix	50–100 cps	5–17 cpi	1 line	Friction and tractor std	90 days	199	Bit image graphics
Legend 1080A	CAL-ABCO	Parallel std; serial opt	Dot matrix	70-140 cps	5-17 cpi	1 line std (4K opt)	Friction and tractor std	90 days	349	Five switch-selectable modes including NLQ
Legend 1380	CAL-ABCO	Parallel std; serial opt	Dot matrix	160 cps	5-17 cpi	2K std; 4K opt	Friction and tractor std	90 days	379	10" version of 1385
Legend 1385	CAL-ABCO	Parallel std; serial opt	Dot matrix	160 cps	5–17 cpi	2K (4K opt)	Friction and tractor std	90 days	449	
A-40	Canon USA, Inc.	Parallel std	Dot matrix	27–140 cps	10 cpi std	1.4K	Friction and tractor std	1 year	349	
A-50	Canon USA, Inc.	Parallel std; serial opt	Dot matrix	34–180 cps	10 cpi std	2K	Friction and tractor std; forms guide opt	1 year	499	Epson FX compatible
A-55	Canon USA, Inc.	Parallel std	Dot matrix	27–140 cps	10 cpi std	1.4K	Friction and tractor std; forms guide opt	1 year	699	

MONITORS

AMDEK

- Video 300 Green ... $119.00
- Video 300A Amber ... $129.00
- Video 310A Amber TTL ... $159.00
- Color 600 Hi-Res. RGB ... $399.00
- Color 722 Dual Mode ... $529.00
- Color 725 ... CALL
- Color 730 ... CALL

MAGNAVOX

- 8562 RGB/Composite ... $279.00
- 613 TTL Green ... $99.99
- 623 TTL Amber ... $99.99

NEC

- JB1205A ... $79.99
- JB1270G/1275A ... (ea.) $99.99
- JB1280G TTL Green ... $129.00
- JB1285A TTL Amber ... $129.00
- JC1401 Multi Sync RGB ... CALL

PRINCETON

- MAX-12 Amber ... $179.00
- HX-9 9" RGB ... $469.00
- HX-9E Enhanced ... $519.00
- HX-12 12" RGB ... $469.00
- HX-12E Enhanced ... $559.00
- SR-12 Hi-Res ... $599.00
- SR-12P Professional ... $699.00

QUADRAM

- 8400 Quadchrome I ... $499.00
- 8410 Quadchrome II ... $339.00
- 8420 Amberchrome ... $179.00
- 8500 Quad Screen ... $1449.00

TAXAN

- 115 12" Green ... $119.00
- 116 12" Amber ... $129.00
- 121 TTL Green ... $139.00
- 122 TTL Amber ... $149.00
- 220 14" Color Composite ... $179.00
- 620 640x200 RGB ... $439.00
- 630 640x200 RGB ... $489.00
- 640 720x400 RGB ... $539.00

ZENITH

- ZVM 1220 Amber ... $99.99
- ZVM 1230 Green ... $99.99
- ZVM 1240 IBM Amber ... $149.00
- ZVM 135 RGB ... $459.00
- ZVM 1330 RGB ... $459.00
- ZVM 1360 RGB ... CALL
- ZVM 1380 E G Comp ... CALL

INTERFACES

AST

- Multi I/O (Apple II) ... $149.00

DIGITAL DEVICES

- Ape Face (Atari) ... $49.99
- U-Print A (Atari) ... $54.99
- U-A16/Buffer (Atari) ... $74.99
- U-Call Interface (Atari) ... $39.99
- U-Print C (C64) ... $49.99
- P-16 Print Buffer ... $74.99
- U-Print 16 apple IIc ... $89.99

MICRO R & D

- Apple IIc Parallel ... $49.99
- Kaypro 2000 Parallel ... $49.99
- C64/128 ... $59.99

Orange Micro

- Grappler CD (C64) ... $89.99
- Grappler Plus (IIe, IIc) ... $89.99
- Grappler C (IIc) ... $89.99
- Grappler 16K (IIe, II+) ... $139.00

PRACTICAL PERIPHERALS

- Graphcard ... $69.99
- Seriall Card ... $99.99
- Microbuffer II+ 64K ... $169.00

QUADRAM

- Microfazer ... from $139.00
- Efazer (Epson) ... from $79.99

PRINTERS

Canon

- A40,A50,A55 ... CALL
- LBP-8A1 Laser ... CALL

CITIZEN

- MSP-10 (80 col.) ... $279.00
- MSP-15 (132 col.) ... $389.00
- MSP-20 (80 col.) ... $349.00
- MSP-25 (132 col.) ... $509.00

C.ITOH

- Prowriter 7500 ... $169.00
- Prowriter 1550P ... $349.00
- Starwriter 10-30 ... $399.00
- 3500 Tri Printer ... $1499.00

corona

- Lazer LP-300 ... $2799.00

DIABLO

- 620 Daisywheel ... $299.00
- D25 Daisywheel ... $549.00
- 635 Daisywheel ... $1099.00
- D80IF Daisywheel ... CALL

daisywriter

- 2000 ... $699.00

EPSON

- Homewriter 10, LX-80 ... CALL
- FX-85, FX-286, RX-100 ... CALL
- DX-10, DX-20, DX-35 ... CALL
- SQ-2000, Hi-80, HS-80, AP-80 ... CALL
- LQ-800, LQ-1000 ... CALL

JUKI

- 6000 Letter Quality ... CALL
- 6100 Letter Quality ... CALL
- 6200 Letter Quality ... CALL
- 6300 Letter Quality ... CALL
- 6500 Letter Quality ... CALL
- 5510 Dot Matrix Color ... CALL

LEGEND

- 808 Dot Matrix 100 cps ... $179.00
- 1080 Dot Matrix 100 cps ... $259.00
- 1380 Dot Matrix 130 cps ... $289.00
- 1385 Dot Matrix 165 cps ... $339.00

NEC

- 3000 Series ... $779.00
- 8000 Series ... $1099.00
- ELF 360 ... $399.00
- Pinwriter 560 ... $999.00

OKIDATA

- 182, 183, 192, 193, 2410, 84 ... CALL
- Okimate 10 (Specify C64/Atari) $189.00
- Okimate 20 (IBM) ... CALL

Panasonic

- KX1080 ... NEW
- KX1091 ... $259.00
- KX1092 ... $389.00
- KX1592 ... $469.00
- KX1595 ... $659.00

QUADRAM

- Quadjet ... $399.00
- Quad Laser ... CALL

SILVER-REED

- 500 Letter Quality ... $219.00
- 550 Letter Quality ... $419.00
- 800 Letter Quality ... $699.00

star

- SG-10C (C64 Interface) ... CALL
- SB/SD/SG/SR Series ... CALL
- Powertype Letter Quality ... CALL

Texas Instruments

- TI850 ... $529.00
- TI855 ... $639.00
- TI865 ... $799.00

TOSHIBA

- 1340 (80 column) ... $369.00
- P341 (132 column) ... $799.00
- P351 (132 column) ... $1049.00

SOFTWARE FOR IBM

ANSA SOFTWARE

- Paradox ... $499.00

ASHTON-TATE

- Framework II ... $389.00
- dBase III Plus ... $389.00

BATTERIES INCLUDED

- Isgur Portfolio ... $159.00

BORLAND

- Lightening ... $59.99
- Sidekick (unprotected) ... $57.99
- Reflex ... $59.99
- Travelling Sidekick ... $44.99

CENTRAL POINT

- Copy II PC-Backup ... $29.99
- PC Option Board ... $84.99

DECISION RESOURCES

- Chartmaster ... $229.00
- Signmaster ... $159.00
- Diagram Master ... $209.00

FIFTH GENERATION

- Fast Back ... 99.99

FUNK SOFTWARE

- Sideways ... $44.99

HARVARD SOFTWARE INC.

- Total Project Manager ... $269.00
- Presentation Graphics ... $239.00

INFOCOM

- Cornerstone ... CALL

LIFETREE

- Volkswriter III ... $159.00

LIVING VIDEOTEXT

- Think Tank ... $109.00
- Ready ... $64.99

LOTUS

- Symphony ... CALL
- 1-2-3 Version 2 ... CALL

MECA SOFTWARE

- Managing Your Money 2.0 ... $99.99
- Manage Your Market ... $89.99

MICROPRO

- Easy ... $94.99
- WordStar 2000 ... $239.00
- WordStar 2000+ ... $289.00
- WordStar Professional ... $189.00

MICRORIM SOFTWARE

- R:Base 4000 ... $249.00
- R:Base 5000 ... $389.00
- Clout 2.0 ... $129.00

MICROSOFT

- Flight Simulator ... $34.99
- MultiPlan ... $129.00
- Word ... $249.00
- Mouse ... $139.00

MICROSTUF

- Crosstalk XVI ... $89.99
- Crosstalk Mark IV ... $149.00
- Remote ... $89.99

MULTIMATE

- Multi Mate Word Proc ... $219.00
- Advantage ... $289.00
- On File ... $89.99
- Just Write ... $89.99

NOUNEMON

- Intuit ... $69.99

NORTON

- Norton Utilities 3.1 ... $57.99

ONE STEP

- Golf's Best ... $34.99

PFS: IBM

- Proof ... $59.99
- File/Graph ... (ea.) $84.99
- Report ... $74.99
- Write/Proof Combo ... $84.99

PROFESSIONAL SOFTWARE

- Write-N-Spell ... $89.99

THE SOFTWARE GROUP

- Enable ... $329.00

SATELLITE SYSTEMS

- Word Perfect 4.1 ... $219.00

SORCIM/IUS

- Accounting AP/AR/GL/INV/OE ... (ea.) $299.00
- SuperCalc III ... $199.00
- EasyWriter II System ... $239.00
- Super Project ... $199.00

SUBLOGIC

- Jet ... $37.99

IBM

IBM PC SYSTEMS

Configured to your specifications.

Call for Best Price!

IBM-PC, IBM-XT, IBM-AT

AT&T

- Safari (7300) ... CALL
- 6300 ... CALL

corona

- PPC400 Dual Portable ... $1289.00
- PPCXT 10 meg Portable ... $1989.00
- PC40022 Dual Desktop ... $1389.00
- PC400-HD2 10 meg ... $1989.00

ITT

- ITT X-TRA
  - 256K, 2 Drive System ... CALL
  - 256K,10 meg Hard Drive System CALL
  - XP5, 20 meg ... CALL

KAYPRO

- KP-2000 Portable ... CALL
- Kaypro PC ... CALL

SANYO

MBC 550-2, MBC 555-2, MBC 675 Portable, MBC775, MBC 880 Desktop CALL

SPERRY

- Sperry-AT ... as low as $1749.00
- Sperry-IT ... as low as $2699.00
- Call for Specific Configuration!
- All Models ... CALL

ZENITH

PC-138 Series, PC-148 Series, PC-158 Series, PC-160 Series, PC-171 Series, AT-200 Series ... CALL

MULTIFUNCTION CARDS

AST

- RamVantage ... $349.00
- Rampage-PC ... $379.00
- Rampage-AT ... CALL
- Six Pack Plus ... $229.00
- I/O Plus II ... $139.00
- Advantage-AT ... $399.00
- Preview Mono ... $299.00
- PC Net Cards ... $379.00
- 5251/11 On-line ... $669.00
- 5251/12 Remote ... $579.00

dca

- IRMA 3270 ... $879.00
- IRMA Print ... $999.00
- IRMA Smart Alec ... $779.00

EVEREX

- Edge Card ... $259.00
- Graphics Edge ... $219.00
- Magic Card II ... $159.00
- Magic Card I ... $99.99

HERCULES

- Graphics ... $299.00
- Color ... $159.00

IDEAssociates

- IDEA 5251 ... $549.00

INTEL

- PCNC8087 5MHz
- PCNC8087-2 8 MHz ... CALL
- PCNC80287 6 MHz ... FOR
- 1010 PC-Above Board ... YOUR
- 1110 PS-Above Board ... PC
- 2010 AT-Above Board

MYLEX

- The Chairman ... $439.00

PARADISE

- Color/Mono Card ... $149.00
- Modular Graphics Card ... CALL
- Multi Display Card ... $199.00
- Five Pack C, S ... $99.99
- High Res Mono ... CALL

PERSYST

- Bob Board ... $359.00

QUADRAM

- Quadport-AT ... $119.00
- Liberty-AT (128K) ... $349.00
- The Gold Quadboard ... $449.00
- The Silver Quadboard ... $239.00
- Expanded Quadboard ... $199.00
- Liberty ... $309.00
- QuadSprint ... $499.00
- QuadLink ... $399.00
- QuadColor ... $199.00
- Quadboard-AT ... $399.00
- 8600 E.G.A. card ... $399.00

TECMAR

- Captain - 64 ... $199.00
- Graphics Master ... $469.00

COMPUTER MAIL ORDER

1-800-233-8950 DEPT. A906

Model Name	Manufacturer/ Distributor	Compatibility	Print Technology	Speed	Pitch	Buffer	Feed Type	Warranty	Suggested Retail Price	Comments
BJ-80	Canon USA, Inc.	Parallel std	Bubble jet	110–220 cps	10 cpi std	2–4.3K	Pin std	1 year	599	Cross between ink jet and thermal transfer
PJ-1080A	Canon USA, Inc.	Parallel std	Ink jet (color)	37 cps	12 cpi std	1 line	Friction std	1 year	699	
H-80	Centronics	Parallel std; serial opt	Dot matrix	27–160 cps	5–20 cpi	2K opt	Cut sheet and fan fold std	1 year	699	
Citizen 120D	Citizen America Corp.	Parallel std; serial opt	Dot matrix	25-120 cps	5-20 cpi	4K	Tractor std	1 year	249	
MSP-10	Citizen America Corp.	Parallel std; serial opt	Dot matrix	40–160 cps	10–12 cpi	1K	Friction and tractor std; cut sheet feeder opt	18 months	399	Emulates IBM graphics
MSP-15	Citizen America Corp.	Parallel std; serial opt	Dot matrix	40–160 cps	10 cpi	1K	Tractor and friction std; cut sheet feeder opt	18 months	549	
MSP-20	Citizen America Corp.	Parallel std; serial opt	Dot matrix	50–100 cps	10–12 cpi	8K	Friction and tractor std; cut sheet feeder opt	18 months	499	Can create own graphics
Premiere 35	Citizen America Corp.	Parallel std; serial opt	Daisy wheel	35 cps	10-15 cpi	8K	Tractor std	1 year	599	Diablo print wheel
MPS 1000	Commodore Business Machines	Parallel and serial std	Dot matrix	20–100 cps	12–17 cpi	1K	Friction and tractor std	90 days	299.95	
Printelex	Computer Peripherals	Parallel and serial std	Dot matrix	160 cps	12 cpi	1 line	Friction std	90 days	157.50	40 column, comes with power pack; 1 1/2 pounds
Dataproducts 8010	Dataproducts Corp.	Parallel and serial std	Dot matrix	30–180 cps	10–17 cpi	2K (8K opt)	Friction and tractor std	1 year	499	Block or dot addressable graphics
Dataproducts 8012	Dataproducts Corp.	Parallel std	Dot matrix	20–180 cps	10–17 cpi	2K std (8K opt)	Friction and tractor std	1 year	499	Block or dot addressable graphics
FORTIS DX-15XL	Dynax	Parallel std; serial opt	Daisy wheel	20 cps	10–15 cpi	5K	Friction std; cut sheet feeder and tractor opt	90 days	599	
FORTIS DX-25	Dynax	Parallel and serial std	Daisy wheel	25 cps	10–15 cpi	7K	Friction and tractor std; cut sheet feeder opt	90 days	645.95	
AP-80	Epson America, Inc.	Apple exclusive	Dot matrix	15–75 cps	9-17 cpi	1K	Friction and tractor std; auto sheet feeder opt	1 year	379	
DX-10	Epson America, Inc.	Parallel std	Daisy wheel	10 cps	10–12 cpi	None	Friction, tractor and cut sheet feeder opt	1 year	299	
DX-20	Epson America, Inc.	Diablo all purpose interface std	Daisy wheel	20 cps	10-15 cpi	1K (7K opt)	Friction std; tractor and cut sheet feeder opt	1 year	459	110 column
FX-85	Epson America, Inc.	Parallel std	Dot matrix	32–160 cps	N/A	8K	Friction and tractor std; cut sheet feeder opt	1 year	549	
FX-286	Epson America, Inc.	Parallel std	Dot matrix	40-200 cps	5-20 cpi	8K	Friction and tractor std; cut sheet feeder opt	1 year	799	
Homewriter 10	Epson America, Inc.	PIC's (Printer Interface Cartridges) are available for most home computers)	Dot matrix	16–100 cps	5-20	1K	Friction std; tractor and cut sheet opt	1 year	249	Selectype available
HS-80	Epson America, Inc.	Parallel std	Ink jet	32–160 cps	5-20 cpi	1K	Friction; auto cut sheet feeder opt	1 year	499	
JX-80	Epson America, Inc.	Parallel std	Dot matrix	160 cps	10–12 cpi	2K	Friction and tractor std	1 year	399	
LQ800	Epson America, Inc.	Parallel and serial std	Dot matrix	60-180 cps	10-15 cpi and proportional	7K	Friction std	1 year	799	24-pin printhead
LX-80	Epson America, Inc.	Parallel std	Dot matrix	16–100 cps	4.3-20 cpi	1K (32K opt)	Friction std; tractor and cut sheet feeder opt	1 year	329	Over 160 typestyles available through Selectype
LX-90	Epson America, Inc.	PIC	Dot matrix	16–100 cps	5-20 cpi	1K	Friction and tractor std	1 year	329	
CP-80 Type 1	Everett/Charles Marketing Service, Inc.	Parallel std; serial opt	Dot matrix	80 cps	N/A	None	Friction and tractor std	90 days	175	Bit image graphics; Superscript
Facit 4509	Facit, Inc.	Parallel std	Dot matrix	70–120 cps	10–17 cpi and proportional	None	Tractor std	90 days	425	IBM compatible graphics
Facit 4510	Facit, Inc.	Parallel and serial std	Dot matrix	70–120 cps	10–17 cpi and proportional	2K	Friction and tractor std	90 days	495	Block and pin addressable graphics
Facit 4511	Facit, Inc.	Parallel and serial std	Dot matrix	40–160 cps	10–17 cpi and proportional	2K	Friction and tractor std	90 days	595	Wide carriage version $795
Facit 4513	Facit, Inc.	Parallel and serial std	Dot matrix	40–160 cps	10–17 cpi and proportional	2K	Friction and tractor std	90 days	695	
Facit D2000	Facit, Inc.	Parallel or serial std	Daisy wheel	24–30 cps	10–15 cpi and proportional	2K	Friction std; tractor opt	90 days	695	
DX2100	Fujitsu America, Inc.	Parallel std; serial opt	Dot matrix	220 cps	10–17 cpi	2K (18K opt)	Friction and tractor std; cut sheet feeder opt	1 year	495 (Color 645)	Dot addressable graphics
DX2200	Fujitsu America, Inc.	Parallel std; serial opt	Dot matrix	220 cps	10–17 cpi	7K std (16K opt)	Friction and tractor std; cut sheet feeder opt.	1 year	645	

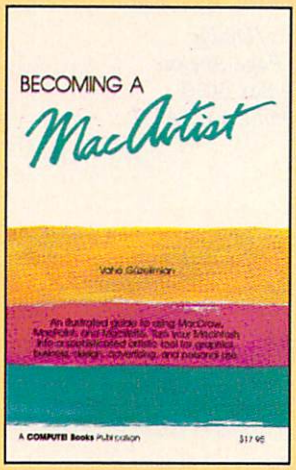
Model Name	Manufacturer/ Distributor	Compatibility	Print Technology	Speed	Pitch	Buffer	Feed Type	Warranty	Suggested Retail Price	Comments
GE 8100	General Electric	Parallel std; Atari, Commodore and IBM PCjr interfaces available	Thermal transfer (non-impact dot matrix)	25–50 cps	24 cpi	2K	Friction std	2 years	259.95	Block graphics; special graphics characters
MP-1300AI	Hattori Seiko	Parallel and serial std	Dot matrix	300 cps	10-20 cpi	10K (7K when downloadable characters)	Friction and tractor std	2 years	799	Optional color printer kit for under $200
SP-1000	Hattori Seiko	Parallel or serial std	Dot matrix	100 cps	10–17 cpi and proportional	Varies with model	Friction and tractor std	2 years	299	Commodore/VIC version: $270
Thinkjet (HP2225)	Hewlett-Packard	Parallel, HP-1B, HP-1L and serial available	Thermal ink jet	150 cps	5–18 cpi	1K	Friction and pin std	1 year	495	Best results using special paper; portable
Juki 5510	Juki Office Machines Corp.	Parallel std	Dot matrix	20–180 cps	10–17 cpi	3K (15K opt)	Friction and tractor std	1 year	499	
Juki 6000	Juki Office Machines Corp.	Parallel std; serial opt	Daisy wheel	10 cps	10–15 cpi	2K-serial	Friction std	1 year	295	
Juki 6100	Juki Office Machines Corp.	Parallel std; serial opt	Daisy wheel	17 cps	10–15 cpi and proportional	2K	Friction std; tractor and cut sheet feeder opt	N/A	599	
Juki 6200	Juki Office Machines Corp.	Parallel std; serial opt	Daisy wheel	30 cps	10–15 cpi and proportional	3K std (15K opt)	Friction std; tractor and cut sheet feeder opt	1 year	745	
Westrex One	Litton Westrex	Parallel or serial std	Dot matrix	42–140 cps	10–15 cpi	2K	Friction and pin std	1 year	199	
MT85	Mannesmann Tally	Parallel, serial or Apple std	Dot matrix	45–180 cps	10–17 cpi	3K	Friction and tractor std	1 year	499	Dot addressable graphics
MT 86	Mannesmann Tally	Parallel, serial, or Apple std	Dot matrix	45–180 cps	10–17 cpi	N/A	Friction and tractor std	1 year	599	
PC-PR105A	NEC Home Electronics, Inc.	Parallel std	Dot matrix	46–92 cps	10 cpi	4K	Friction and tractor std	90 days labor; 1 year parts	399	
E.L.F. 350	NEC Information Systems, Inc.	Parallel and serial std	Daisy wheel	19 cps	10–15 cpi	2K	Cut sheet guide std; cut sheet feeder and tractor opt	1 year	545	
E.L.F. 360	NEC Information Systems, Inc.	Parallel and serial std	Daisy wheel	19 cps	10–15 cpi	2K	Cut sheet guide std; cut sheet feeder and tractor opt	1 year	545	
P2	NEC Information Systems, Inc.	None std; IBM serial, serial, IBM parallel, and parallel opt	Dot matrix	35–180 cps	10–17 cpi	3.5K–5K	Tractor std; cut sheet feeder opt	1 year	699	
Microline 182	Okidata	Parallel std; serial opt	Dot matrix	30-120 cps	5–17 cpi	1 line	Friction and pin std	1 year	299	Block and bit image graphics
Microline 182 TTY	Okidata	Parallel std; serial opt	Dot matrix	30-120 cps	5-17 cpi	1 line	Friction and pin std	1 year	349	Designed for communications applications; 4 levels of intelligence
Microline 183	Okidata	Parallel std;serial opt	Dot matrix	30-120 cps	5–17 cpi	1 line	Friction and tractor std	1 year	549	Wide carriage version of 183; block and bit image graphics
Microline 192	Okidata	Parallel std; serial opt	Dot matrix	33–160 cps	5–17 cpi	8K	Friction and pin std	1 year	499	Block and bit image graphics
Microline 193	Okidata	Parallel and serial std	Dot matrix	33–160 cps	5–17 cpi	8K	Friction and tractor std	1 year	699	Wide carriage version of 192; block and bit image graphics
Microline 292	Okidata	Parallel or serial std	Dot matrix	100-200 cps	10-17 cpi	8K	Pin std; tractor and cut sheet feeder opt	1 year	699	
Okidata 120	Okidata	Commodore serial	Dot matrix	30-120 cps	5–17 cpi	1 line	Friction and pin std	1 year	269	All points addressable graphics; Commodore Special Graphics
Okimate 10	Okidata	Commodore and Atari	Dot matrix	60 cps	5–17 cpi	1 line	Friction and pin std	90 days	208	All points addressable graphics; Commodore Special Graphics
Okimate 20	Okidata	IBM, Apple and Amiga	Dot matrix	40–80 cps	5–17 cpi	8K	Friction and tractor std	90 days	268	High resolution, all points addressable, bit image graphics
KX-P1080	Panasonic Co.	Parallel std; serial opt	Dot matrix	20–100 cps	10–17 cpi	1K	Friction and tractor std	2 years	319	Bit image graphics; can emulate Image Writer; Epson RX-80 compatible; color ribbons available
KX-P1091	Panasonic Co.	Parallel std; serial opt	Dot matrix	29–120 cps	10–17 cpi	1K (4K opt)	Friction and tractor std	2 years	399	Same as above
KX-P1092	Panasonic Co.	Parallel std; serial opt	Dot matrix	22–180 cps	10–12 cpi and proportional	7K	Friction and push tractor std	2 years	499	Bit image graphics; Epson FX-80 compatible; color ribbons available
KX-P1592	Panasonic Co.	Parallel std; serial opt	Dot matrix	38–180 cps	10–17 cpi and proportional	7K text std (32K opt)	Push tractor or friction std	2 years	699	Colored ribbon available
KX-P3131	Panasonic Co.	Parallel std; serial opt	Daisy wheel	17 cps	N/A	6K (32K opt)	Friction std; tractor and auto cut sheet feeder opt	2 years	419	Diablo 630 code compatible; color ribbons available
KX-P3151	Panasonic Co.	Parallel std; serial opt	Daisy wheel	22 cps	10–12 cpi	7K std (54.5K opt)	Friction std; tractor and cut sheet feeder opt	2 years	659	

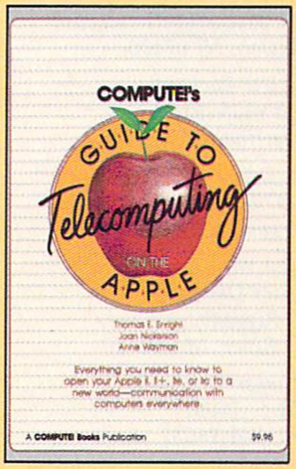
Model Name	Manufacturer/ Distributor	Compatibility	Print Technology	Speed	Pitch	Buffer	Feed Type	Warranty	Suggested Retail Price	Comments
RP2200Q	Ricoh Corporation	Parallel and serial std	Daisy wheel	20-22 cps	10-15 cpi	One line	Friction std; auto sheet feeder and tractor opt	90 days	699	
Letter Master	Royal Consumer Business Products	Parallel std; serial opt	Daisy wheel	10 cps	10–12 cpi	84 characters	Friction std	90 days labor; 1 year parts	349.95	
OfficeMaster 2000	Royal Consumer Business Products	Parallel and serial std	Daisy wheel	23 cps	10–15 cpi and proportional	1.5K	Friction std; tractor and sheet feeder opt	90 days labor; 1 year parts	599.95	
SP-1200 PLUS	Sakata	Parallel std	Dot matrix	120 cps	5-16.5	N/A	Friction and pin std	90 days	399	
SP-1500	Sakata	Parallel std	Dot matrix	180 cps	5-16.5	N/A	N/A	90 days	489	
SP-5500	Sakata	Parallel std; serial opt	Dot matrix	36-180 cps	5-18 cpi	3K	Friction and tractor std	90 days	699	Wide carriage
EXP 500	Silver-Reed, Inc.	Parallel or serial std	Daisy wheel	14 cps	10–15 cpi	None	Friction std; tractor and cut sheet feeder opt	90 days	449	
EXP 550	Silver-Reed, Inc.	Parallel or serial std	Daisy wheel	19 cps	10–15 cpi and proportional	None	Friction std; cut sheet feeder and tractor opt	90 days	649	
EXP 600	Silver-Reed, Inc.	Parallel or serial std	Daisy wheel	25 cps	10-15 cpi and proportional	3K std (19K and 40K opt)	Friction std; tractor and sheet feed opt	90 days labor; 1 year parts	N/A	June release
Silver-Reed 400	Silver-Reed, Inc.	Parallel or serial std	Daisy wheel	10 cps	10–15 cpi	None	Friction std; tractor opt	90 days	249	
NX-10	Star Micronics	Parallel std	Dot matrix	30-120 cps	5-10 cpi	5K	Friction and tractor std	1 year	349	Emulates IBM graphics printer
Powertype	Star Micronics	Parallel std; serial opt	Daisy wheel	18 cps	10–15 cpi and proportional	1 line	Friction std; tractor opt	180 days	499	
SD-10	Star Micronics	Parallel std; serial opt	Dot matrix	160 cps	N/A	2K	Friction and tractor std	1 year	449	Ultra-high resolution bit image graphics
SD-15	Star Micronics	Parallel std; serial opt	Dot matrix	160 cps	N/A	16K	Friction and tractor std	1 year	599	
SR-10	Star Micronics	Parallel std; serial opt	Dot matrix	200 cps	N/A	2K	Friction, tractor, automatic single sheet feed std	1 year	649	
SR-15	Star Micronics	Parallel std; serial opt	Dot matrix	200 cps	N/A	2K	Friction, tractor, single sheet feeder std	1 year	799	15" carriage
STX-80	Star Micronics	Parallel std	Thermal transfer dot matrix	60 cps	5–11 cpi	1 line	Friction std	1 year	199	Dot addressable, bit image graphics
Compumate 2100	Swintec Corporation	Parallel and serial std	Daisy wheel	16 cps	10–15 cpi	1 line	Friction std	90 days	599	
M-20P	TAB Products	Parallel std	Dot matrix	120 cps	10–15 cpi	1K	Friction and pin tractor opt	90 days	299	Dot addressable graphics
M-21P	TAB Products	Serial std	Dot matrix	120 cps	10–15 cpi	1K	Friction and pin tractor opt	90 days	378	Dot addressable graphics
M-22P	TAB Products	Parallel std	Dot matrix	160 cps	10–15 cpi	1K	Friction and pin std; tractor and cut sheet feeder opt	90 days	499	Dot addressable graphics
M-23S	TAB Products	Serial std	Dot matrix	160 cps	10–15 cpi	1K	Pin and friction std; cutsheet feeder and tractor opt	90 days	549	
M-24P	TAB Products	Parallel std	Dot matrix	160 cps	10–15 cpi	1K	Tractor and friction std; cut sheet feeder opt	90 days	699	
CGP 220	Tandy Corp.	Parallel and serial std	Ink jet	37 cps	12 cpi	1 line	Friction std	90 days	599	
DMP 105	Tandy Corp.	Parallel and serial std	Dot matrix	80 cps	10–17 cpi and proportional	N/A	Friction and tractor std	90 days	199.95	Bit image graphics
DMP 130	Tandy Corp.	Parallel and serial std	Dot matrix	100 cps	10–16 cpi	N/A	Friction and tractor std	90 days	349.95	Bit image graphics
DWP 220	Tandy Corp.	Parallel and serial std	Daisy wheel	20 cps	10–12 and proportional	N/A	Friction std; tractor opt	90 days	599	
TRP 100	Tandy Corp.	Parallel and serial std	Dot matrix	50 cps	N/A	N/A	Friction and sheet feed std	90 days	299.95	Uses thermal paper
IMP-24	Weigh-Tronix, Inc.	Parallel or serial std	Dot matrix	16.8 cps	N/A	1 line	Friction std	90 days	135 (24 cpl) 150 (32 or 40 cpl)	Dot addressable graphics
Companion 10I	Xerox/Diablo	Parallel std	Dot matrix	80 cps	10–15 cpi	1K	Friction and tractor std	90 days	399	Bit map graphics
Companion 12CQ1	Xerox/Diablo	Parallel std	Dot matrix	60–150 cps	10–15 cpi	2K	Friction and tractor std	90 days	599	

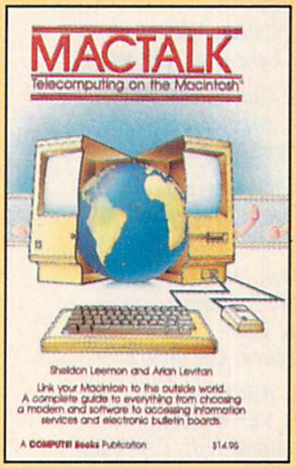
APPLE TITLES

BACK BY POPULAR DEMAND!

These popular titles for the Apple-II series and Macintosh computers contain expert programming advice, in-depth tutorials, valuable buying guides, challenging business, home, and educational applications, and exciting games. They're all written in COMPUTE!'s easy-to-understand style to provide you with hours of creative entertainment and information.







Becoming a MacArtist
Vahé Guzelimian, 312 pages
An illustrated guide to Macintosh graphics with which you can explore the full power and flexibility of *MacDraw*, *MacPaint*, and *MacWrite*.
$17.95 ISBN 0-942386-80-9

COMPUTE!'s Apple Games for Kids
Clark and Kathy H. Kidd, 305 pages
An instant library of educational software including simple tests, music and graphics programs, a typing tutor, menu planner, and game-writing utilities for the whole family.
$12.95 ISBN 0-942386-91-4

COMPUTE!'s Guide to Telecomputing on the Apple
Thomas E. Enright, Joan Nickerson, and Anne Wayman, 173 pages
An informative and easy-to-understand guide to telecomputing on the Apple, from selecting hardware and software to accessing large databases.
$9.95 ISBN 0-942386-98-1

COMPUTE!'s Kids and the Apple
Edward H. Carlson, 255 pages
An entertaining and easy-to-use book for kids ages 10–14 interested in learning BASIC programming on their Apple computers.
$12.95 ISBN 0-942386-76-0

MacTalk: Telecomputing on the Macintosh
Sheldon Leemon and Arlan Levitan, 263 pages
A complete guide to telecomputing on the Macintosh, from choosing a modem and software to accessing information services and electronic bulletin boards.
$14.95 ISBN 0-942386-85-X

COMPUTE!'s First Book of Apple
Edited, 228 pages
A collection of 35 exciting games, educational programs, home applications, and graphics routines for owners of Apple II, IIe, and IIc computers.
$12.95 ISBN 0-942386-69-8

Easy BASIC Programs for the Apple
Brian Flynn, 390 pages
A wide-ranging selection of short BASIC programs—for home, school, and office—for Apple II-series computers.
$14.95 ISBN 0-942386-88-4

Pick your favorite titles and order today!
Call toll-free 800-346-6767 (in NY 212-887-8525).
Or mail your payment with $2.00 shipping and handling per book to COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Please allow 4–6 weeks for delivery.

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

COMPUTE! books are available in the U.K., Europe, the Middle East, and Africa from Holt Saunders, Ltd., 1 St. Anne's Road, Eastbourne, East Sussex BN21 3UN, England, and in Canada from McGraw-Hill, Ryerson Ltd., 330 Progress Ave., Scarborough, Ontario, Canada M1P 2Z5.

For more information on any of the printers listed here, please contact:

Alphacom
2323 S. Bascom Avenue
Campbell, CA 95008

Apple Computer
20525 Mariani Avenue
Cupertino, CA 95014

Aprotek
1071-A Avenida Acaso
Camarillo, CA 93010

Axiom
1014 Griswold Avenue
San Fernando, CA 91340

Axonix
417 Wakara Way
Salt Lake City, UT 84108

Blue Chip Electronics
2 W. Alameda Drive
Tempe, AZ 85282

Brother International
8 Corporate Place
Piscataway, NJ 08854

C. Itoh Digital Products
19750 S. Vermont Avenue
Suite 220
Torrance, CA 90502

CAL-ABCO
6041 Variel Avenue
Woodland Hill, CA 91367

Canon USA
System Division
One Canon Plaza
Lake Success, NY 11042

Centronics Data Computer
1 Wall Street
Hudson, NH 03051

Citizen America
2425 Colorado Avenue #300
Santa Monica, CA 90404

Commodore Business Machines
1200 Wilson Drive
West Chester, PA 19380

Computer Peripherals
6400 Canoga Avenue
Suite 305
Woodland Hills, CA 91367

Dataproducts
6200 Canoga Avenue
Woodland Hills, CA 91365

Dynax
6070 Rickenbacker Road
Commerce, CA 90040

Epson America
3415 Kashiwa Street
Torrance, CA 90505

Everett/Charles Marketing Services
6101 Cherry Avenue
Fontana, CA 92335

Facit
9 Executive Park Drive
P.O. Box 334
Merrimack, NH 03054

Fujitsu America
3055 Orchard Road
San Jose, CA 95134

General Electric
Electronics Park
Syracuse, NY 13221

Hattori Seiko Computer Peripherals
Division of Hattori Corporation of America
1111 Macarthur Boulevard
Mahwah, NJ 07430

Hewlett-Packard
1100 Wolfe Road
Cupertino, CA 95014

Juki Office Machine
20437 S. Western Avenue
Torrance, CA 90501

Litton Westrex
Printer Division
100 Riggenback Road
Fall River, MA 02720

Mannesmann Tally
8301 S. 180th
Kent, WA 98032

NEC Home Electronics (USA)
Personal Computer Division
Elk Grove Village, IL 60007

NEC Information Systems
1414 Massachusetts Avenue
Boxborough, MA 01719

Okidata
532 Fellowship Road
Mt. Laurel, NJ 08054

Panasonic
Computer Products Division
One Panasonic Way
Secaucus, NJ 07094

Ricoh America
5 Dedrick Place
West Caldwell, NJ 07006

Royal Consumer Business Products
500 Day Hill Road
Windsor, CT 06095

Sakata USA
651 Bonnie Lane
Elk Grove Village, IL 60007

Silver-Reed America
19600 S. Vermont Avenue
Torrance, CA 90502

Star Micronics Peripheral Division
200 Park Avenue
Suite 2309
Pan Am Bldg.
New York, NY 10166

Swintec
23 Poplar Street
P.O. Box 421
East Rutherford, NJ 07073

TAB Products
1400 Page Mill Road
Palo Alto, CA 94304

Tandy /Radio Shack
1800 One Tandy Center
Fort Worth, TX 76102

Weigh-Tronix
1000 N. Armstrong
Fairmont, MN 56031

Xerox/Diablo
901 Page Avenue
P.O. Box 5030
Fremont, CA 94537

©

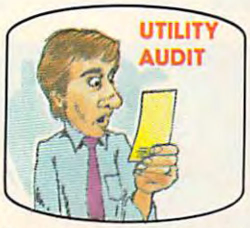
Get the jump on the weatherman by accurately forecasting the local weather yourself!

A scientifically proven way to develop an awesome memory.

You are trapped in a five-story, 125-room structure made entirely of ice. Find the exit before you freeze!

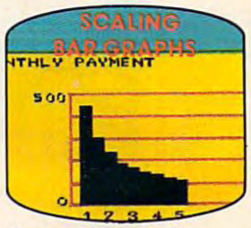
Take control of your personal finances in less than one hour a month.

The beautiful princess is held captive by deadly dragons. Only a knight in shining armor can save her now!



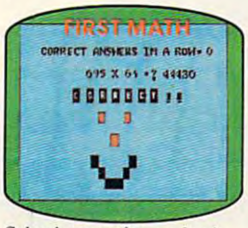
Cut your energy costs by monitoring your phone, electric and gas bills.

Computerize car maintenance to improve auto performance, economy and resale value.

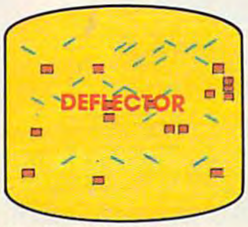


Create multi-colored bar graphs with a surprisingly small amount of memory.

A time-saving organizer for coupons, receipts and more.



School-age and pre-school children are rewarded for right answers, corrected on their wrong ones.



A real brainflexer. Deflect random balls into targets on a constantly changing playfield.

A fun way to dramatically increase typing speed and accuracy.

Get up to 20 new programs and games every month in COMPUTE!

Every month, COMPUTE! readers enjoy up to 20 brand-new, ready-to-run computer programs, even arcade-quality games.

And when you subscribe to COMPUTE!, you'll get them all for less than 15 cents each!

You'll find programs to help you conserve time, energy and money. Programs like Cash Flow Manager, Retirement Planner, Coupon Filer, Dynamic Bookkeeping.

You'll enjoy games like Air Defense, Boggler, Slalom, and High Speed Mazer.

Your children will find learning fast and fun with First Math, Guess That Animal, and Mystery Spell.

Looking for a challenge? You can write your own games. Customize BASIC programs. Even make beautiful computer music and pictures.

It's all in COMPUTE!. All ready to type in and run on your Atari, Apple, Commodore, TI-99/4A, IBM PC, or PCjr computers.

What's more, you get information-packed articles, product reviews, ideas and advice that add power and excitement to all your home computing.

And when it's time to shop for peripherals or hardware, check COMPUTE! first. Our product evaluations can save you money and costly mistakes. We'll even help you decide what to buy: Dot-matrix or daisy-wheel printer? Tape storage or disk drive? What about modems? Memory expansion kits? What's new in joysticks, paddles, and track balls?

Order now! Mail the postpaid card attached to this ad and start receiving every issue of COMPUTE!.

For Faster Service
Call Toll-Free
1-800-247-5470
(in Iowa
1-800-532-1272)

COMPUTE! P.O. Box 10954, Des Moines, IA 50340

MIAMI ICE

Jeff Kulczycki

Here's an action game that challenges both your driving skills and powers of concentration. Originally written for the Commodore 128 with a disk drive, "Miami Ice" has been translated to work on the Commodore 64, Atari 400/800/XL/XE (with at least 32K RAM), and Apple II-series computers. A joystick is required.

Ah, Miami—sun city of the South. A sparkling metropolis blessed with a tropical climate, palm trees, beaches, revived art deco architecture, stylish pastels, and classy elegance. Almost paradise.

You wake up on another bright, sunny Miami morning, sip a glass of freshly squeezed orange juice, don your white linen suit and sunglasses, and stroll outside—then get the shock of your life.

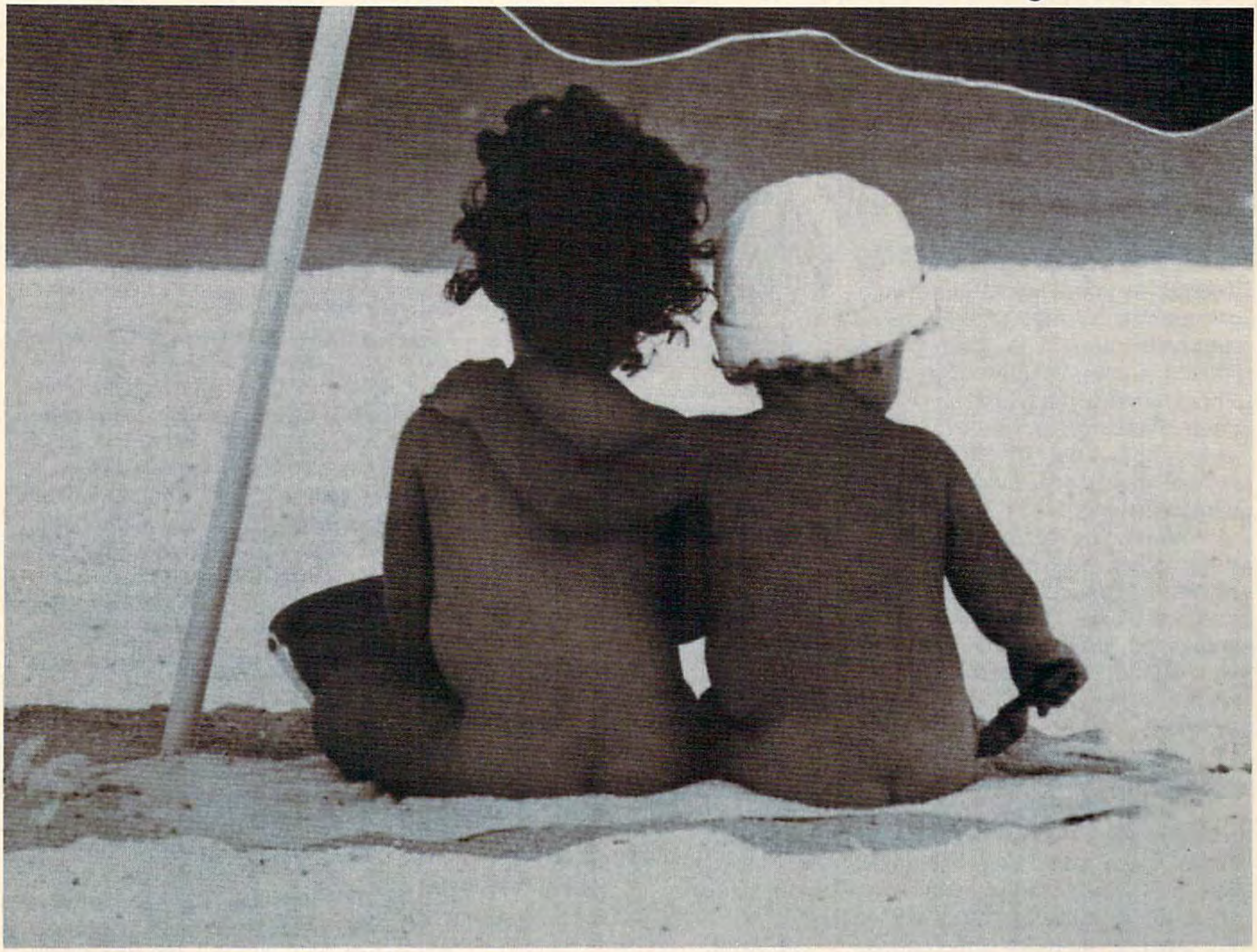
What's going on here? Overnight, a freak shift in the jet streams has piped a blistering cold front down from Ohio. The weatherman had predicted a brief shower last evening, but that's not what happened. Instead, the Florida peninsula was blasted by the worst ice storm in 400 years. The Everglades are frozen solid. The pink flamingos are blue. And the streets of Miami are coated with a shimmering layer of slippery ice.

As you start your car—the pampered engine coughs and sputters in the bitter cold—you wonder what it's going to be like driving to work. A Miami native, you've never driven on ice before. In fact, you've never even *seen* this much ice since your boss's retirement party last year, when the caterers made that life-size ice sculpture of Ponce de Leon. You've heard the horror stories told by tourists about winter driving conditions up North, but never thought it could happen to you—not here, in Miami.

The minute you pull out onto the street, your worst fears come true. When you step on the gas pedal, the wheels spin and the car accelerates sluggishly. When you turn the steering wheel, the car slides all over the road. And when you step on the brakes—well, forget it.

You realize, desperately, that you've got to make it to the parking garage across town without smashing your car to smithereens. It won't be easy. But at least there's one thing in your favor—you've got the whole road to yourself.

The Source Is Friendly.℠



Many online information services claim to be "user friendly." But only one really lives up to that promise.

The Source.

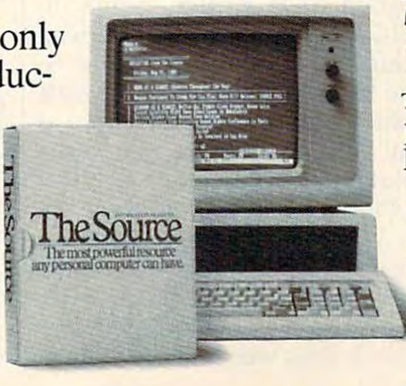
You see, The Source is specifically designed to save you time online. With new, shorter menus. Simpler commands. And a user's manual so well-written and easy to understand, it's setting an industry standard.

The Source is also the only service that gives you introductory lessons and assistance, free of on-line charges. So you get up-to-speed on our dime, not yours.

You get to the information you need in record time, without frustration. Everything from the hour's headlines to travel reservations. From special interest groups to online stock trading. So you don't waste your valuable time. Or money.

Call 1-800-336-3366, send the coupon, or visit your nearest computer dealer.

And make friends with America's friendliest online information network.



The Source is a service mark of Source Telecomputing Corp., a subsidiary of The Reader's Digest Assn., Inc. © 1985 Source Telecomputing Corp.

INFORMATION NETWORK

The Source℠

The most powerful resource any personal computer can have.

I'd like to make friends with The Source. Please send more information to:

Name ______________________

7826J24

Address ______________________

City, State ____________ Zip ________

Mail to: Source Telecomputing Corporation, 1616 Anderson Road, McLean, VA 22102

Everyone else, it seems, had the good sense to stay home.

Out Of Control

Despite minor variations, all four versions of "Miami Ice" work basically the same. Using a joystick, you have to drive your car over ice-covered streets to reach the safety of a garage. The joystick button is the gas pedal, and pushing the stick right or left steers the car in the corresponding direction.

But here's the twist—the car doesn't respond instantly to your commands. It tends to slide in the same direction even after you've steered it toward another direction. Then, when you try to recover, you often overcorrect and start sliding in yet another new direction. It's an inertial nightmare—much like real winter driving.

When you hit a guardrail or some other obstruction, your car cracks up. You get three cars per game. If you reach the safety of the garage, the game isn't over. Instead, you advance to another screen whose streets are even harder to navigate.

The number of points you score depends on how soon you reach the garage. As an incentive to recklessness, a timer starts counting down when you begin each new screen. If you reach the garage, you score the number of points left on the timer. If the timer runs out, you can still reach the garage, but you won't get any points. However, you will advance to the next screen.

Be sure to read the special instructions for each version before typing in the program and playing the game.

Commodore 128

The 128 version of Miami Ice (Program 1) is written completely in BASIC using BASIC 7.0's excellent sprite commands. It runs as fast and as smoothly as the other versions, which all employ machine language.

Plug a joystick into port 2 and leave a disk in the drive. After each game, if your score ranks you among the top players, the program lets you enter your initials and then saves the high score data to disk.

To complete each level, you merely have to steer your car into the parking garage from any angle. There are a total of four screens, and each screen displays the timer value in the upper-left corner and your current score immediately to the right.

Commodore 64

The 64 version of Miami Ice is written completely in machine language and must be entered with the Commodore 64 version of the "MLX" machine language entry program found elsewhere in this issue. Be sure you read and understand the instructions for using MLX before you begin entering the data from Program 2. When you first run MLX, you'll be asked to supply a starting address and an ending address. Here are the addresses you'll need for Miami Ice:

Starting address: 0801
Ending address: 1320

After entering all the data from Program 1, be sure to save at least one copy before you exit MLX. Although the 64 version of Miami Ice is written in machine language, you start the program as if it were written in BASIC: load the program, then type RUN and press RETURN.

Plug a joystick into port 1. To steer your car safely into the parking garage and advance to the next screen, you have to enter the front of the garage without bumping into the black lines which mark its three walls. Indicators on the screen show the timer value and your current score.

There are seven screens in all. The game normally starts at screen 1, but you can begin a new game at any screen you want by moving the joystick up or down to change the screen number. This lets you skip the easier screens as you become a better player, or peek at the hardest screens while you're still a beginner.

Atari 400/800/XL/XE

The Atari version of Miami Ice (Program 3) is written largely in BASIC, but has an interrupt-driven machine language subroutine to move the car using player/missile graphics. The car itself is composed of all four players to gain more resolution and colors.

Plug a joystick into port 1. To steer your car safely into the parking garage and advance to the next screen, you have to enter the front of the garage without bumping into the black lines which mark its three walls. Indicators on the screen show the timer value and your current score.

There are seven screens in all. You'll notice that some screens have more than one route to the garage. The first time you play the game, it starts at screen 1. Subsequent games begin at the screen where the last game ended. But you can start a new game at any screen you want by moving the joystick up or down to change the screen number.

Apple II Series

The Apple version of Miami Ice is written completely in machine language and must be entered with the Apple version of the "MLX" machine language entry program found elsewhere in this issue. Be sure you read and understand the instructions for using MLX before you begin entering the data from Program 4. When you first run Apple MLX you'll be asked for a starting address and an ending address. Here are the addresses you'll need for Miami Ice:

Starting address: 1000
Ending address: 1597

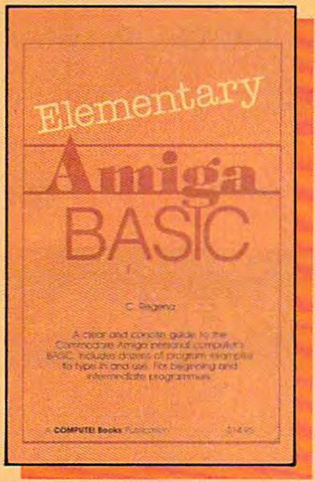
After you have typed in all the data from Program 4, be sure to save at least one copy before you exit MLX. To start MLX, enter BRUN *"filename"* (where *filename* is the name you used when you saved the Miami Ice data with MLX), then press RETURN.

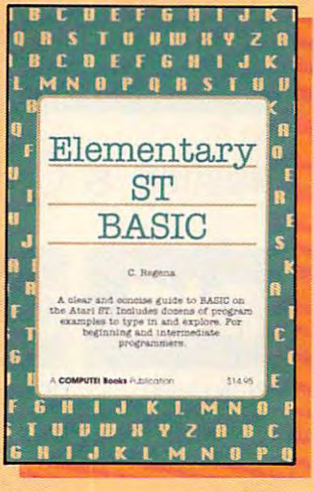
To begin playing, plug in a joystick or paddles. To reach the garage safely and advance to the next screen, you have to enter the front of the garage without bumping into any of its walls. There are seven screens in all. The game normally starts at screen 1, but you can begin a new game at any screen you want by pressing the controller button to change the screen number.

Program 1: Miami Ice For Commodore 128

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
EF 10 OPEN2,8,2,"HI-SCORE,S,W"
      :CLOSE2:OPEN15,8,15:INPU
      T#15,A$,B$:IFB$<>"FILE E
      XISTS"THENCLOSE15:GOSUB7
      50
EM 20 COLOR0,16:COLOR4,11
BR 30 PRINT"{CLR}{RED}{7 DOWN}
```





BASIC programming at its best!

Two new programming guides from COMPUTE! Books.

Elementary Amiga BASIC
C. Regena
0-87455-041-6, $14.95
Disk $15.95

Elementary ST BASIC
C. Regena
0-87455-034-3, $14.95
Disk $15.95

Written by the author of the bestselling *Programmer's Reference Guide to the TI-99/4A*, these books introduce you to the new and powerful BASIC on the Amiga and Atari ST personal computers. The computers' impressive graphics, animation, and sound can be unlocked with the right commands, and BASIC is the place to start. Regena shows you how—in the clear, concise language that's made her such a popular writer.

Complete descriptions of the Amiga's and ST's BASIC commands, syntax, and organization take you from novice to full-power programming. Sample programs and subroutines, all ready to type in, are included. Plus, both books offer you working software while showing you how to write your own programs. *A disk is also available for each book which includes all the programs from the book in an easy, ready-to-load format.*

SPECIAL COMBINATION OFFER
Order the book and disk together for only $29.95!

You'll find these new programming guides and many more useful, entertaining COMPUTE! books at your local computer and book stores. Or you can order directly from COMPUTE! Books.

For the fastest service, call toll free 1-800-346-6767 (in NY 212-887-8525). Or mail the attached coupon with your payment to COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Please add shipping and handling charges to all orders: In U.S., $2.00 per book, disk, or combination package; $5.00 per item airmail.

I want the best in BASIC programming guides. Please send me:

______ ***Elementary ST BASIC,*** (034-3), $14.95 each ______
______ ***Elementary ST BASIC Disk,*** $15.95 each ______
______ ***Elementary ST BASIC*** disk and book combination, $29.95 ______
______ ***Elementary Amiga BASIC,*** (041-6), $14.95 each ______
______ ***Elementary Amiga BASIC Disk,*** $15.95 each ______
______ ***Elementary Amiga BASIC*** disk and book combination, (057-2), $29.95 ______

Subtotal ______
NC residents add 4.5% sales tax ______
Shipping charges ______
($2.00 per item in U.S. and surface mail;
$5.00 per item airmail)
Total amount enclosed ______

ALL ORDERS MUST BE PREPAID IN U.S. FUNDS

☐ Payment enclosed (check or money order)
☐ Charge ☐ MasterCard ☐ Visa ☐ American Express
Account No. ______ Exp. Date ______ (Required)
Name ______
Address ______
City ______
State ______ Zip ______
Allow 4–6 weeks for delivery. 36612111

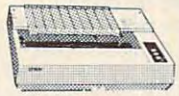
COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

COMPUTE! books are available in the U.K., Europe, the Middle East, and Africa from Holt Saunders, Ltd., 1 St. Anne's Road, Eastbourne, East Sussex BN21 3UN, England and in Canada from McGraw-Hill, Ryerson Ltd., 330 Progress Ave., Scarborough, Ontario, Canada M1P 2Z5.

Lyco Computer Marketing & Consultants

1091 . . . . . $228

SAVE ON THESE IN STOCK PRINTERS



SG-10 . . . . $205

LEGEND
808 148
1080 197
1380 224
1385 268

BROTHER
HR-15XL-P 359
HR-15XL-S 359

SILVER REED
EXP400 249
EXP500 295
EXP550 399
EXP770 749

PANASONIC
1091 228
3131 (NEW) 264
1092 325
3151 409
1080 (NEW) 209
1592 (NEW) 439

EPSON
LX80 209
FX85 333
JX80 .... Call
Homewriter 10 193
DX10 207
DX20 297
DX35 597
AP-80 244
HI-80 355
HS-80 298
FX-286 (NEW) 489
LQ-800 (NEW) 529
LQ-1000 (NEW) 659

SEIKOSHA
SP-1000 VC (C-64) 185
SP-1000 A Centronics 199
SP-1000 I IBM 199
SP-1000 AS RS-232 199
SP1000 AP Apple IIc 199
BP-5200 I 649
BP sheet feeder 199
SP-1000 ribbon 8.50
BP-5200 ribbon 12.50

OKIDATA
Okimate 10 179
182 214
192 348
193 563

STAR MICRONICS
SG-10 205
SG-10c 219
SG-15 367
SD-10 319
SD-15 438
SR-10 469
SR-15 578
SB-10 589
Powertype 297
NX-10 (NEW) .... CALL
NB-15 (NEW) .... CALL

JUKI
Juki 6100 344
RS232 Serial Board 55
6100 Tractor 119
6100 Sheet Feeder 209
Juki 6300 757

DIABLO
D25 549
630 API 1599
630 ECS 1759
D 80 IF 2395
P 32 CQ1 699
P 38 1749
C 150 999

CORONA
LP300 2495
200361 Toner Cartridge 89

CITIZEN
MSP-10 255
MSP-15 355
MSP-20 337
MSP-25 495
120D 188
Premier 35 429

C. ITOH
Prowriter 8510 sp+ .... Call
15505 sp+ .... Call
Printmaster .... Call

COLOR RIBBONS NOW AVAILABLE!!

DUST COVERS

Atari
520ST 11.95
130XE 6.99
800XL 6.99
1050 .... 6.99
1025 .... 7.99

Commodore
C128 .... 7.99
1571/1541 .... 6.99
1902 .... 10.95
1702 .... 8.99
C64/Vic20 .... 6.99

Panasonic
1090/1091 .... 8.99
1092 .... 8.99
1093 .... 9.99

Star Micronics
SG/SD10 .... 8.99
SG/SD15 .... 9.99
SR10 .... 9.99
SR15 .... 9.99

Okidata
82/92 .... 8.99
83/93 .... 9.99
193 .... 9.99

DRIVES

INDUS
GT Atari 195
GT Commodore 195

COMTEL
Enhancer 2000 (C-64) 159

TYMAC
MDD-640 3½"
Apple Drive 640K 289

COMMODORE
1571 .... CALL
1541 .... CALL

INTERFACING

DIGITAL DEVICES
Ape Face XLP (Atari) 49
U-Print A (Atari) 54
U-Print A 16K Buffer 74
U-Print AP 16K (Apple) 99

CARDCO
G-WIZ (C-64) 54
C/?PS (C-64) 49
C/?B (C-64) 39

XETEC
Super Graphix 64 64
Super Graphix JR 64 45

MICROBITS
MPP-1150 (Atari) 54
MPP-1150XL (Atari) 59
MicroPrint (Atari) 39

TYMAC
Connection (C-64) 55
Tackler (Apple) 49
PPC-100 (Apple) 39

MICROTEK
Dumpling GX (Apple) 59
Dumpling 16K (Apple) 89
RV-611C (Apple) 49

ORANGE MICRO
GRAPPLER + (Apple) 85
Grappler 16K (Apple) 149
ORANGE (Apple) 59
Grappler CD (C-64) 79

DISKETTES

DENNISON
ELEPHANT 5¼" SSSD 11.99
ELEPHANT 5¼" SSDD 12.99
ELEPHANT 5¼" DSDD 14.99

SUNKYONG
SKC 5¼" SSDD 11.99
SKC 5¼" DSDD 13.99

MAXELL
5¼" MD1 .... 13.99

VERBATIM
5¼" SSDD 13.99
5¼" DSDD 19.99

BONUS
5¼" SSDD 8.99
5¼" DSDD 12.99

IBM-PC

SSI (IBM)
Battle for Normandy 24.95
Knights of Desert 24.95
Tigers in Snow 24.95
Computer Baseball 24.95
Cartels & Cutthroats 24.95
OP Market Garden 29.95
50 Mission Crush 24.95

MICROPROSE (IBM)
F-15 Strike Eagle 20.75
Solo Flight 20.75
Silent Service 20.75
Decision in Desert 24.95
Crusade Europe 24.95

SUBLOGIC (IBM)
Jet Simulator 34.95
Scenery Disks EA 14.95
Set 1-6 .... 69.95

LEADING EDGE
Nutshell 69.95
Nutshell Filer 149.00

ACTIVISION (IBM)
Borrowed Time 24.75
Mindshadows 24.75
Music Studio 29.95
Alter Ego 29.95

UNISON WORLD (IBM)
Printmaster 34.95
Art Gallery 1 .... 24.95

SYNAPSE (IBM)
Synstock 64.95
Essex 28.95
Wizard of Wall St 28.95
Brimstone 28.95

QUICKVIEW (IBM)
Zoomracks 4.95

BRODERBUND (IBM)
Bank St. Writer 48.95
The Print Shop 34.95
Graphics Library 1 .... 22.95
Ancient Art of War 27.95
Champ Lode Runner 22.95
Karateka 22.95

MICROLEAGUE (IBM)
M L Baseball 24.95
General Mgr 24.95
85 Team Disk 14.95

TOLL FREE 1-800-233-8760

MasterCard

TO ORDER

VISA

CALL TOLL FREE 1-800-233-8760
In PA 717-494-1030
Customer Service 717-494-1670

or send order to
Lyco Computer
P.O. Box 5088
Jersey Shore, PA
17740

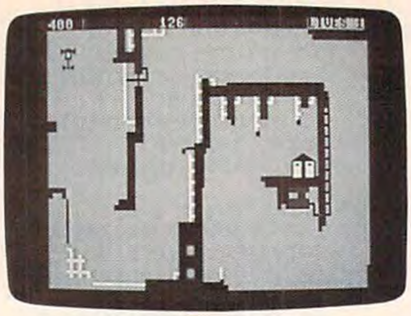
RISK FREE POLICY

In-stock items shipped within 24 hours of order. No deposit on C.O.D. orders. Free shipping on prepaid cash orders within the continental U.S. Volume discounts available. PA residents add sales tax. APO, FPO, and international orders add $5.00 plus 3% for priority mail service. Advertised prices show 4% discount for cash, add 4% for MasterCard or Visa. Personal checks require 4 weeks' clearance before shipping. Ask about UPS Blue and Red label shipping. All merchandise carried under manufacturer's warranty. Free catalog with order. All items subject to change without notice.

```
      {15 RIGHT}{RVS}MIAMI ICE
      "
SD 40 PRINT"{BLU}{DOWN}
      {11 SPACES}JOYSTICK IN P
      ORT 2"
JJ 50 PRINT"{BLK}{DOWN}{BLK}
      {11 SPACES}[LEFT]
      {2 SPACES}TURN LEFT":PRI
      NT"{11 SPACES}[RIGHT] TU
      RN RIGHT"
QC 60 PRINT"{11 SPACES}[FIRE]
      {2 SPACES}ACCELERATE":PR
      INT"{YEL}{DOWN}
      {13 SPACES}READING DATA.
      .."
BD 70 GOSUB1800:PRINT"{UP}
      {BLU}{10 SPACES}PRESS BU
      TTON TO PLAY"
DB 80 IFJOY(2)<>128THEN80
BE 90 HY=3:SC=0:SN=1
XD 100 FAST:ONSNGOSUB760,1020,
       1280,1550:SLOW:PRINT"
       {HOME}"TAB(32);"{BLK}LI
       VES";HY:COLOR0,16:TM=40
       0:T=0:XE=0
KC 110 GOSUB540:
AH 120 POKE2041,62:MOVSPR2,X,Y
       :SPRITE2,1,2,0,0,0,1:PO
       KE2040,57:XE=BUMP(2)
MX 130 MOVSPR1,30#0:SPRITE1,1,
       9,0,0,0,1:SPRCOLOR1,2:M
       OVSPR1,40,65:I=4:AN=180
       :HT=135:TH=0:XE=BUMP(2)
       +BUMP(1)
BQ 140 PRINT"{HOME}{RVS}";TM;"
       {LEFT} {OFF}"
JX 150 IFJOY(2)=0THEN150
PH 160 IFJOY(2)=3THEN280
GC 170 IFJOY(2)=7THEN310
PR 180 IFJOY(2)=128THENMOVSPR1
       ,AN#1:TH=1:SOUND1,5000,
       24,2,1000,3,3
XG 190 POKE2040,53+I:IFBUMP(1)
       =3THEN490:ELSEIFBUMP(2)
       AND1THEN420
PC 200 IFTH>1THENONABS(T-20)GO
       TO410
RC 210 T=T+1
KS 220 IFHT>180THENIFHT-180>AN
       THEN360
RS 230 IFHT>180THENIFHT-180<AN
       THEN370
KR 240 IFHT<180THENIFHT+180<AN
       THEN380
RD 250 IFHT<180THENIFHT+180>AN
       THEN390
MM 260 TM=TM-1
CC 261 IFTM<0THENTM=0
XP 262 PRINT"{HOME}{RVS}";TM;"
       {LEFT} {OFF}"
MH 270 IFJOY(2)<>3THEN300
AQ 280 AN=AN+45:IFAN>360THENAN
       =45
KD 290 I=I-1:IFI=0THENI=8:GOTO
       190
DS 300 IFJOY(2)<>7THEN340
CF 310 AN=AN-45:IFAN<0THENAN=3
       15
KE 320 IFAN=360THENAN=0
DD 330 I=I+1:IFI=9THENI=1:GOTO
       190
HH 340 IFJOY(2)=128THENSOUND1,
       5000,24,2,1000,3,3:TH=T
       H+1:T=0:IFTH>15THENTH=1
       5:GOTO190
CP 350 GOTO190
FG 360 HT=HT+((AN+(360-HT))/10
       ):MOVSPR1,HT#TH:IFHT<36
       0THEN260:ELSE:HT=0:GOTO
       390
RX 370 HT=HT-((HT-AN)/10):MOVS
       PR1,HT#TH:GOTO260
SM 380 HT=HT-((HT+(360-AN))/10
       ):MOVSPR1,HT#TH:IFHT>0T
       HEN260:ELSE:HT=360:GOTO
       370
JQ 390 HT=HT+((AN-HT)/10):MOVS
       PR1,HT#TH:GOTO260
QR 400 GOTO260
XD 410 TH=TH-1:T=0:IFTH<1THENT
       H=1:GOTO220:ELSE220
KF 420 POKE2040,63:FORDELAY=1T
       O150:NEXT:SPRITE 1,0
RJ 430 SOUND1,2000,100,0,1000,
       1,3,100
JB 440 T=0:HY=HY-1:PRINT"
       {HOME}"TAB(32)"LIVES";H
       Y:IFHY=0THEN460
CD 450 XE=BUMP(2):SLEEP2:POKE2
       040,57:MOVSPR1,33,55:XE
       =BUMP(2):GOTO130
HX 460 SLEEP2:PRINT"{9 DOWN}
       {15 RIGHT}{RVS}{BLK}GAM
       E OVER{OFF}"
PD 470 OPEN2,8,2,"HI-SCORE,S,R
       ":INPUT#2,A$,B$:CLOSE2:
       IFSC>VAL(A$)THENFORI=1T
       O8:SPRITEI,0:NEXT:GOTO5
       60
HK 480 IFJOY(2)<>128THEN480:EL
       SE:XE=BUMP(2):GOTO90
AS 490 POKE53280,6:MOVSPR1,40#
       0:PLAY"QGRGRG":IFTM=0TH
       EN520
GP 492 SD=INT(2000/TM)
XG 500 FORTY=1TOTMSTEP5:PRINT"
       {HOME}{RVS}";TM-TY:PRIN
       T"{HOME}{RVS}";TAB(13);
       SC+TY:IFTM-TY<99THENIFT
       M-TY>90THENPRINT"{HOME}
       {RVS}{4 SPACES}{OFF}"
HM 510 SOUND1,3000+(SD*TY),1:N
       EXT:SC=SC+TM:PRINT"
       {HOME}{RVS}{2 SPACES}0
       {SPACE}{OFF}";TAB(13);"
       {RVS}";SC
PC 520 SLEEP1:SN=SN+1:IFSN=5TH
       ENSN=1
CB 530 GOTO100
QK 540 PRINT"{HOME}{RVS}";TM;"
       {HOME}{RVS}"TAB(13);SC:
       RETURN
XB 550 REM ********** HI SCORE
       **********
GB 560 PLAY"O4SCCFGBBAR AB":PR
       INT"{CLR}{2 DOWN}
       {11 SPACES}YOUR SCORE:
       {SPACE}";SC:AB=65:OP=0
JG 570 PRINT"{10 SPACES}CCCCCC
       CCCCCCCCCCCCCC{UP}"
PB 580 OPEN2,8,2,"HI-SCORE,S,R
       ":FORI=1TO10:INPUT#2,B$
       (I):INPUT#2,A$(I):NEXT:
       CLOSE2:SCRATCH"HI-SCORE
       "
GB 590 FORU=1TO10:IFSC>VAL(B$(
       U))THENNEXT
FX 600 U=U-1:FORE=1TOU-1:A$(E)
       =A$(E+1):B$(E)=B$(E+1):
       NEXT:B$(U)=RIGHT$(STR$(
       SC),LEN(STR$(SC))-1):A$
       (U)="---"
PB 610 TE=LEN(B$(U)):FORP=1TO6
       -TE:B$(U)="0"+B$(U):NEX
       T
FG 620 PRINT"{2 DOWN}":FORE=10
```

```
        TO2STEP-1:PRINTTAB(11);
        11-E;"{2 SPACES}";A$(E)
        ;"{3 SPACES}";B$(E):NEX
        T
SF 630 PRINTTAB(10);10;"
        {2 SPACES}";A$(1);"
        {3 SPACES}";B$(1)
EQ 640 PRINT"{HOME}{5 DOWN}":F
        ORI=1TO11-U:PRINT:NEXT:
        NM$=""
CM 650 PRINT"{UP}"TAB(16+OP);C
        HR$(AB)
GF 660 IFJOY(2)=7THENAB=AB-1:I
        FAB<65THENAB=65:GOTO650
PK 670 IFJOY(2)=3THENAB=AB+1:I
        FAB>90THENAB=90:GOTO650
BD 680 IFJOY(2)=128THENNM$=NM$
        +CHR$(AB):AB=65:OP=OP+1
        :SLEEP1:IFOP=3THEN700
XA 690 GOTO650
ER 700 A$(U)=NM$:OPEN2,8,2,"HI
        -SCORE,S,W":FORI=1TO10:
        PRINT#2,B$(I):PRINT#2,A
        $(I):NEXT:CLOSE2
GF 710 PRINT"{HOME}":FORI=1TO1
        8:PRINT:NEXT
MG 720 PRINT"{7 SPACES}PRESS B
        UTTON TO PLAY AGAIN":GO
        TO480
FM 740 REM ******* CLEAR HI-SC
        ORES *******
GH 750 SCRATCH"HI-SCORE":PRINT
        "{CLR}MAKING HI-SCORE":
        OPEN2,8,2,"HI-SCORE,S,W
        ":FORI=1TO10:PRINT#2,"0
        00000":PRINT#2,"---":NE
        XT:CLOSE2:RETURN
DB 760 X=62:Y=135:COLOR4,16
KG 770 PRINT"{CLR}{RVS}{RED}
        {28 SPACES}ƐC3"
JX 780 PRINT"{RVS} {OFF}
        {24 SPACES}{WHT}++{RED}
        ƐC3{RVS}{9 SPACES}ƐC3
        {OFF}"
KG 790 PRINT"{RVS} {OFF}
        {25 SPACES}{WHT}ƐQ3
        {5 SPACES}ƐQ3+ƐW3{RED}
        ƐC3{RVS}{3 SPACES}ƐC3
        {OFF}"
AH 800 PRINT"{RVS} {OFF}
        {31 SPACES}Ɛ53{RVS}
        {3 SPACES}{OFF}
        {2 SPACES}{RED}ƐC3{RVS}
        {2 SPACES}{OFF}"
CD 810 PRINT"{RVS} {OFF}
        {37 SPACES}{RVS}
        {2 SPACES}{OFF}"
FC 820 PRINT"{RVS} ƐC3{OFF}
        {36 SPACES}ƐC3{RVS}
        {OFF}"
EX 830 PRINT"{RVS}{9 SPACES}
        ƐC3{OFF}{29 SPACES}
        {RVS}{2 SPACES}{OFF}
        {WHT}++++{RED}
        {3 SPACES}ƐC3{RVS}
        {4 SPACES}ƐC3{OFF}
        {24 SPACES}{BLK}ƐQ3
        {RED}{RVS} {OFF}"
HS 840 PRINT"{RVS} {OFF}{WHT}+
        ++{RED}{5 SPACES}{WHT}+
        {RED}ƐC3{RVS}
        {15 SPACES}ƐC3{OFF}
        {11 SPACES}{BLK}ƐQ3
        {RED}{RVS} {OFF}"
CM 850 PRINT"{RVS} {OFF}
        {2 SPACES}{WHT}+
        {5 SPACES}ƐZ3++{RED}
        {RVS}ƐF3{3 SPACES}{OFF}
        {WHT}ƐW3{4 SPACES}+++
        {RED}ƐC3{RVS}{2 SPACES}
        {OFF}{11 SPACES}{BLK}
        ƐQ3{RED}{RVS} {OFF}"
HE 860 PRINT"{RVS} {OFF}
        {9 SPACES}{WHT}ƐQ3++
        {RED}{RVS}{2 SPACES}
        {OFF}{WHT}+ƐW3
        {5 SPACES}ƐQ3++{RED}ƐC3
        {RVS} ƐC3{OFF}
        {10 SPACES}ƐD3{RVS}
        {OFF}"
GB 870 PRINT"{RVS} {OFF}
        {10 SPACES}{WHT}ƐZ3ƐX3
        {RED}{RVS} {OFF}{WHT}+
        ƐW3{8 SPACES}ƐQ3+{RED}
        {RVS}{2 SPACES}{OFF}
        {9 SPACES}{BLK}ƐQ3{RED}
        {RVS}{2 SPACES}{OFF}"
EK 880 PRINT"{RVS} {OFF}
        {12 SPACES}{RVS} {OFF}
        {WHT}ƐW3{10 SPACES}ƐQ3
        {RED}ƐC3{RVS} {OFF}
        {9 SPACES}{BLK}ƐQ3{RED}
        {RVS}{2 SPACES}{OFF}"
AJ 890 PRINT"{RVS} {OFF}
        {12 SPACES}ƐT3
        {13 SPACES}{RVS} {OFF}
        {9 SPACES}{BLK}ƐQ3{RED}
        {RVS}{2 SPACES}{OFF}"
PQ 900 PRINT"{RVS} {OFF}
        {26 SPACES}{RVS} {OFF}
        {9 SPACES}{BLK}ƐQ3{RED}
        {RVS}{2 SPACES}{OFF}"
GE 910 PRINT"{RVS} {OFF}
        {37 SPACES}ƐC3{RVS}
        {OFF}"
CM 920 PRINT"{RVS} {OFF}
        {37 SPACES}{BLK}ƐQ3
        {RED}{RVS} {OFF}"
GR 930 PRINT"{RVS} {OFF}
```

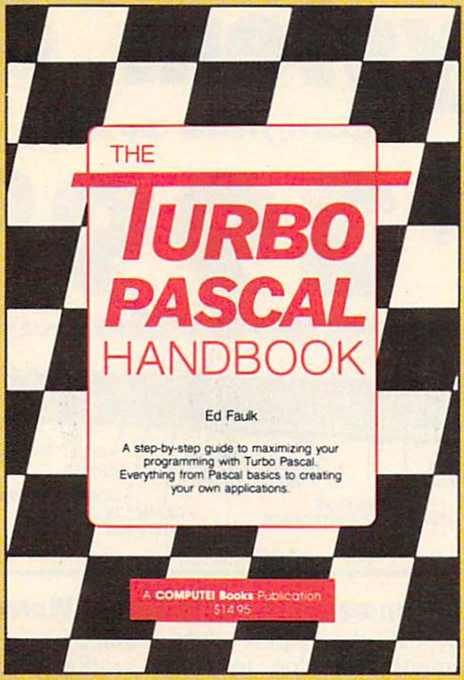
```
{37 SPACES}{BLK}EQ3
{RED}{RVS} {OFF}"
MQ 940 PRINT"{RVS} {OFF}
{37 SPACES}{BLK}EQ3
{RED}{RVS} {OFF}"
EK 950 PRINT"{RVS} {OFF}EF3
{11 SPACES}E@3
{24 SPACES}{BLK}EQ3
{RED}{RVS} {OFF}"
DQ 960 PRINT"{RVS}{2 SPACES}
{OFF}EF3{10 SPACES}
{RVS} {OFF} {WHT}EQ3
{22 SPACES}{BLK}EQ3
{RED}{RVS} {OFF}"
MK 970 PRINT"{RVS}{3 SPACES}
{OFF}{10 SPACES}{RVS}
{OFF}{WHT}+EW3
{15 SPACES}EA3
{6 SPACES}{RED}ED3{RVS}
{OFF}"
SD 980 PRINT"{RVS}{3 SPACES}
{OFF}EF3{WHT}E2 R3+ES3
{4 SPACES}{RED}ED3{RVS}
{OFF}{WHT}+++
{12 SPACES}EA3+EW3
{5 SPACES}{RED}{RVS}EV3
{2 SPACES}{OFF}"
SJ 990 PRINT"{RVS}{4 SPACES}
{OFF}EF3{WHT}+++ES3
{RED} ED3{RVS}
{3 SPACES}EC3{OFF}{WHT}
EQ3++E2 R3+{6 SPACES}
EQ3+++ES3{2 SPACES}
{RED}{RVS}EV3{4 SPACES}
{OFF}"
PQ 1000 PRINT"{RVS}{39 SPACES}
{OFF}";:POKE2023,224:P
OKE56295,2
DS 1010 RETURN
CC 1020 X=262:Y=142:COLOR4,3
KS 1030 PRINT"{CLR}E13{RVS}
{40 SPACES}{OFF}"
KQ 1040 PRINT"E13{RVS} {OFF}
{8 SPACES}{RVS} {OFF}
{WHT}EQ3E13{RVS} {OFF}
{WHT}CEE3EX3
{17 SPACES}E13{RVS}EF3
{7 SPACES}{OFF}"
XM 1050 PRINT"{RVS} {OFF}
{8 SPACES}{RVS} {OFF}
{WHT}EQ3E13{RVS} {OFF}
{27 SPACES}{RVS}EJ3
{OFF}{7 SPACES}{RED}
ED3E13{RVS}{3 SPACES}
{OFF}{27 SPACES}{RVS}
EJ3{OFF}"
AB 1060 PRINT"{RVS} {OFF}
{8 SPACES}{WHT}EM3 E13
{RVS} {OFF}{BLK}ES3
{26 SPACES}E13{RVS}EJ3
{OFF}"
CG 1070 PRINT"{RVS} {OFF}
{8 SPACES}{WHT}EM3
{BLK}C+EX3{5 SPACES}
{WHT}EQ3{RED}{RVS}EU3
{OFF}{WHT}ES3 {RED}
E12 P3{5 SPACES}E13
{RVS}EJ3{OFF}"
XC 1080 PRINT"{RVS} {OFF}
{8 SPACES}{WHT}EM3
{RED}{RVS} {OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} E13
{14 SPACES}{OFF}{RED}
EF3{4 SPACES}E13{RVS}
EJ3{OFF}"
DB 1090 PRINT"{RVS} {OFF}
{8 SPACES}{WHT}EM3
{RED} {RVS} {OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} {OFF} {WHT}
EQ3E13{RVS} {OFF}{WHT}
EX3 EQ3E13{RVS} {OFF}
{WHT}EX3{2 SPACES}EQ3
```



"Miami Ice," Commodore 128 version: Driving through the streets of Miami isn't easy when they're covered with a layer of ice.

```
E13{RVS} {OFF}{WHT}EX3
E13{RVS} {BLK}EL3{OFF}
{4 SPACES}E13{RVS}EJ3
{OFF}"
CS 1100 PRINT"{RVS} {OFF}
{8 SPACES}{WHT}EM3
{RED} {RVS} {OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} {OFF} {WHT}
EQ3E13{RVS} {OFF}
{2 SPACES}{WHT}EQ3E13
{RVS} {OFF}{3 SPACES}
{WHT}EQ3E13{2 SPACES}
{RVS} {OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
KX 1110 PRINT"{RVS}{2 SPACES}
{OFF}{8 SPACES}{WHT}
EY3{RED}{RVS} {OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} {OFF}
{2 SPACES}{RED}EC3
{2 SPACES}{WHT}EQ3
{4 SPACES}{RED}EC3
{2 SPACES}E13{RVS}
{OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
XR 1120 PRINT"{RVS} {OFF}
{9 SPACES}{RED}{RVS}
{2 SPACES}{OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} {OFF}
{5 SPACES}EC3
{7 SPACES}E13{RVS}
{OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
JS 1130 PRINT"{RVS} {OFF}
{9 SPACES}{RED}{RVS}
{OFF}{5 SPACES}{RED}
ED3C{RVS} ED3{OFF}
{13 SPACES}E13{RVS}
{OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
RQ 1140 PRINT"{RVS} {OFF}
{9 SPACES}{RED}{RVS}
{OFF}{6 SPACES}{WHT}
EQ3{RED}{RVS} {OFF}EH3
{13 SPACES}E13{RVS}
{OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
AR 1150 PRINT"{RVS} {OFF}
{9 SPACES}{RED}{RVS}
{OFF}{6 SPACES}{WHT}
EQ3{RED}{RVS} {OFF}EH3
{13 SPACES}E13{RVS}
{OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
RJ 1160 PRINT"{RVS} {OFF}
{9 SPACES}{RED}{RVS}
{OFF}{6 SPACES}{WHT}
EQ3{RED}{RVS} {OFF}EH3
{6 SPACES}E13{RVS}EF3
{7 SPACES}{OFF}{BLK}
EW3{4 SPACES}E13{RVS}
EJ3{OFF}"
FR 1170 PRINT"{RVS} {BLK}ED3
EF3{OFF}{7 SPACES}
{RED}{RVS} {OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} {OFF}
{9 SPACES}E13{RVS}
{BLU}ED3EF3ED3{OFF}
E13{RVS} {OFF}{BLK}EW3
{4 SPACES}E13{RVS}EJ3
{OFF}"
XJ 1180 PRINT"{RVS} E83
{2 SPACES}{OFF}{BLK}
EG3{4 SPACES}{RED}ED3
{RVS}{2 SPACES}{OFF}
{6 SPACES}{WHT}EQ3
{RED}{RVS} {OFF}
{9 SPACES}E13{RVS}ED3
{BLU}{3 SPACES}{OFF}
{BLK}ES3E13{RVS} {OFF}
{BLK}EW3{4 SPACES}E13
{RVS}EJ3{OFF}"
BC 1190 PRINT"{RVS} E83ED3EF3
{OFF}{BLK}EG3
{13 SPACES}{WHT}EQ3
{RED}{RVS} {OFF}
{10 SPACES}{WHT}E3 T3
{BLK}EZ3E13{RVS} {OFF}
{BLK}EX3{4 SPACES}E13
{RVS}EJ3{OFF}"
QM 1200 PRINT"{RVS} E83
{2 SPACES}{OFF}{BLK}
EG3{12 SPACES}{RED}
{RVS}{3 SPACES}{OFF}
{14 SPACES}E13{RVS}ED3
{OFF}{5 SPACES}{RVS}
EJ3{OFF}"
HS 1210 PRINT"{RVS} E83ED3EF3
{OFF}{BLK}EG3
{12 SPACES}{RED}{RVS}
{3 SPACES}{OFF}
{20 SPACES}E13{RVS}EJ3
{OFF}"
ER 1220 PRINT"{RVS} E83ED3EF3
{OFF}{WHT}ES3
{12 SPACES}{RED}{RVS}
{SPACE}E33Q{RED} {OFF}
{20 SPACES}E13{RVS}EJ3
{OFF}"
DG 1230 PRINT"{RVS} E83ED3EF3
{OFF}{WHT}++
{11 SPACES}{RED}{RVS}
{3 SPACES}{OFF}
{20 SPACES}E13{RVS}EJ3
{OFF}"
ER 1240 PRINT"{RVS} E83
{2 SPACES}{OFF}{WHT}++
ES3{9 SPACES}EA3{RED}
{RVS} E33Q{RED} {OFF}
{SPACE}{WHT} EH3
{17 SPACES}E13{RVS}
{OFF}"
PK 1250 PRINT"{RVS} E83
{5 SPACES}{OFF}{WHT}CC
CCCCE13{RVS}{7 SPACES}
{RED}E2 Y3EC3{OFF}
{17 SPACES}E13{RVS}
{OFF}"
MR 1260 PRINT"{RVS}{39 SPACES}
{OFF}";:POKE2023,224:P
OKE56295,8
XB 1270 RETURN
BA 1280 X=280:Y=200:COLOR4,16
DK 1290 PRINT"{CLR}{GRN}{RVS}
{40 SPACES}{OFF}"
DA 1300 PRINT"{RVS} {OFF}
{19 SPACES}E*3{RVS}
```

Explore Pascal with



THE TURBO PASCAL HANDBOOK

from COMPUTE!.

The Turbo Pascal Handbook
Edward P. Faulk
With *The Turbo Pascal Handbook* and *Turbo Pascal* from Borland International, you'll be gently guided, step-by-step, until you're creating your own powerful applications in this impressive computer language.
$14.95 ISBN 0-87455-037-8

This information-packed book from COMPUTE! is an outstanding resource and programming guide. And it's written in COMPUTE!'s bestselling style so that even beginning programmers can quickly and easily understand all the applications.

Ask for *The Turbo Pascal Handbook* at your local computer store or bookstore. Or order directly from COMPUTE!. Call toll free 1-800-346-6767 (in NY 212-887-8525) or mail the attached coupon with your payment (plus $2.00 shipping and handling per book) to COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Note: You'll need *Turbo Pascal* in order to use this book. The software is not included with *The Turbo Pascal Handbook.*

Yes! Send me _____ copies of *The Turbo Pascal Handbook* at $14.95 each.
My payment is enclosed.

ALL ORDERS MUST BE PREPAID IN U.S. FUNDS

Subtotal _____
NC residents add 4.5% sales tax _____
Shipping and handling _____
($2.00 per book in U.S. and surface mail; $5.00 per book airmail.)
Total enclosed _____

☐ Payment enclosed (check or money order)
☐ Charge ☐ Visa ☐ MasterCard ☐ American Express

Account No. _____ Exp. Date _____ (Required)
Name _____
Address _____
City _____ State _____ Zip _____

Please allow 4–6 weeks for delivery. 36603711

COMPUTE!™ Publications, Inc. abc
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

240K Apple® Compatible Computer System

NEW SALE

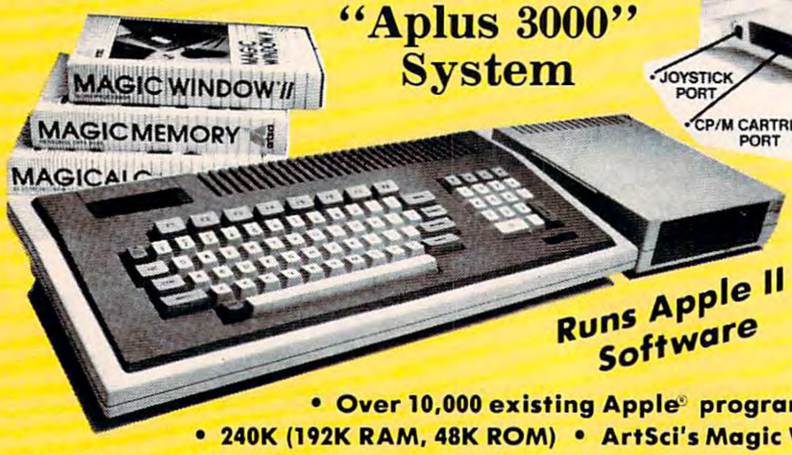
APlus 3000 computer system includes 192K RAM, 48K ROM (32K Microsoft Basic plus 16K ROM Emulator), 160K Laser 5¼" Disk Drive (Runs Apple II Software), Magic Window Wordprocessor, MagiCalc spreadsheet, Magic Memory Database. All for only $399.00

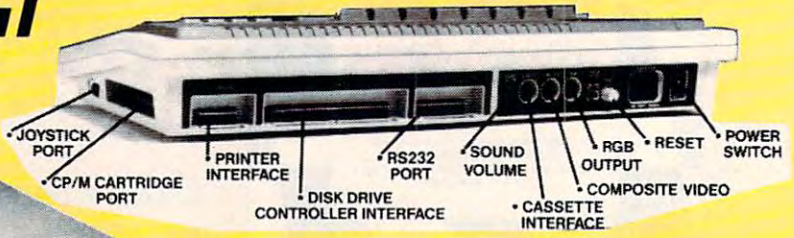
Complete System

$399^{00}

• *15 Day Free Trial*

"Aplus 3000" System





Double Immediate Replacement Warranty

If any of the Aplus 3000 computer system equipment fails due to faulty workmanship or material within 180 days of purchase we will REPLACE it immediately with no service charge!!

- **Over 10,000 existing Apple® programs**
- **Centronics printer interface included**
- **240K (192K RAM, 48K ROM)**
- **ArtSci's Magic Window II, Magic Memory, and MagiCalc included**
- **160K Laser 5¼" Disk Drive (Runs Apple II software)**
- **RGB (80 columns in color) and composite included**

SPECIFICATIONS

A plus 3000 is a complete, self-contained computer based on the popular 6502A microprocessor and can tap into the tremendous software library of Apple II. Features include 192K Bytes RAM, 32KB Enhanced Microsoft BASIC, 80 column text, 560H X 192V color graphic display, 81 key sculptured keyboard and high efficiency switching power supply. Also included as standard are Centronics bus printer interface, Cassette interface, 4 channel sound generator, and 5¼" Apple Compatible Disk Drive.

• TEXT

— 40 columns X 24 rows or 80 columns X 24 rows software selectable.
— 5 X 7 characters in 7 X 8 matrix.
— Upper and lower case characters.
— One of Eight colors for characters/graphics and background, Red, Green, Blue, Cyan, Magenta, Yellow, Black and White.
— Character set with normal, inverse and flashing capabilities.

• GRAPHICS

— 280H X 192V 6 colors — Black, White, Violet, Green, Blue, Orange.
— 280H X 192V 8 colors bit image — Black, White, Red, Green, Blue, Cyan, Magenta, Yellow.
— 560H X 192V 6 colors — Black, White, Violet, Green, Blue, Orange. (High resolution color monitor required)

More Features than Apple® for less than Commodore®

Features	**Aplus 3000**	Apple IIe	Commodore C-128
RAM	**192K**	64K	128K
Runs Apple II Software	**Yes**	Yes	No
Function Keys	**24**	None	16
4 Voice, 6 Octave Sound	**Yes**	No	Yes
Composite Video	**Yes**	Yes	Yes
Disk Drive	**Included**	Extra Cost	Extra Cost
Numeric Keypad	**Included**	Extra Cost	Included
Video Cable	**Included**	Extra Cost	Extra Cost
RGB Color Card	**Included**	Extra Cost	Included
80 Column Card	**Included**	Extra Cost	Included
Centronics Printer Interface	**Included**	Extra Cost	Extra Cost
Drive Controller	**Included**	Extra Cost	Included
$150 Wordprocessor (Magic Window)	**Included**	Extra Cost	Extra Cost
$150 Spreadsheet (MagiCalc)	**Included**	Extra Cost	Extra Cost
$60 Database prg. (Magic Memory)	**Included**	Extra Cost	Extra Cost
Your Cost	**$399.00**	$1745.00	$1117.90

Super Apple Compatible Disk Drive Sale $149.95. Quieter, Cooler, Better Disk Drives for your Apple II plus, IIe, IIc (specify when ordering). List $299.95. Sale $149.95.

15 Day Free Trial — If it doesn't meet your expectations within 15 days of receipt, just send it back to us UPS prepaid and we will refund your purchase price!!

ACCESSORIES	**LIST**	**SALE**
2nd Disk Drive	$299.95	$149.95
2 professional analog joysticks	$ 39.95	$ 24.95
Z-80 cart. allows CP/M use	$ 99.95	$ 59.95
RS232 adapter	$ 99.95	$ 59.95
R/F Modulator (TV hookup)	$ 29.95	$ 19.95
RGB cable (RGB Monitor hookup)	$ 24.95	$ 19.95
Centronics cable (for Centronics printer)	$ 34.95	$ 24.95
Technical reference manual	$ 29.95	$ 19.95
80 columns Hi-Res Green Monitor	$199.00	$ 79.95
80 column Hi-Res RGB Monitor	$399.00	$259.00

Add $25.00 for shipping and handling!!

Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery. 2 to 7 days for phone orders. 1 day express mail! We accept Visa and MasterCard. We ship C.O.D. to continental U.S. addresses only. Add $10 more if C.O.D.

COMPUTER DIRECT

We Love Our Customers

22292 N. Pepper Rd., Barrington, Ill. 60010

312/382-5050 to order

ATARI®

152K *Lowest Price In The USA!* 152K

Computer System Sale

• Students • Word Processing • Home • Business

Limited Time
FREE Software w/ System Purchase*
Home Filing Mgr., Star Raiders, Music Composer, Defender

LOOK AT ALL YOU GET FOR ONLY **$379** **SYSTEM PRICE**

LIMITED QUANTITIES

	LIST PRICE	INDIVIDUAL SALE PRICE
① Atari 130XE 152K Computer	$249.00	$134.95
② Atari 1050 127K Disk Drive	299.00	159.95
③ Atari 1027 Letter Quality 20 CPS Printer	299.00	159.95
Atari Writer Plus Word Processer with Spell Checker	59.95	49.95
Atari BASIC Tutorial Manual	16.95	7.95
TOTALS	**$923.90**	**$512.75**

All connecting cables & T.V. interface included.
☆ Monitors sold separetly.

SAVE OVER $100
All 5 ONLY
$379.00
SYSTEM SALE PRICE

CALL FOR 1027 PRINTER REPLACEMENT OPTIONS

* Free software subject to substitution for other titles

Other Accessories

	List	Sale	
☆ 12" Hi Resolution Green Screen Monitor	$199.00	**$79.95**	Add $9.95 for Connection Cables
☆ 13" Hi Resolution Color Monitor	$399.00	**$149.95**	Add $10 for UPS

15 DAY FREE TRIAL. We give you 15 days to try out this ATARI COMPUTER SYSTEM!! If it doesn't meet your expectations, just send it back to us prepaid and we will refund your purchase price!! **90 DAY IMMEDIATE REPLACEMENT WARRANTY.** If any of the ATARI COMPUTER SYSTEM equipment or programs fail due to faulty workmanship or material within 90 days of purchase we will replace it IMMEDIATELY with no service charge!!

Best Prices • Over 1000 Programs and 500 Accessories Available • Best Service • One Day Express Mail • Programming Knowledge • Technical Support

Add $25.00 for shipping and handling!!
Enclose Cashiers Check, Money Order or Personal Check. Allow 14 days for delivery. 2 to 7 days for phone orders. 1 day express mail! We accept Visa and MasterCard. We ship C.O.D. to continental U.S. addresses only. Add $10 more if C.O.D., add $25 if Air Mail.

COMPUTER DIRECT
We Love Our Customers
22292 N. Pepper Rd., Barrington, Ill. 60010
312/382-5050 to order

COMMODORE 64 COMPUTER

(Order Now)

$139^95

- **C128 Disks 79¢ ea.***
- **Paperbock Writer 64 $39.95**
- **10" Comstar 10X Printer $148.00**
- **13" Color Monitor $149.95**

CALL BEFORE YOU ORDER

COMMODORE 64 SYSTEM SALE

Commodore 64 Plus $30.00 S&H

Com. 1541 Disk Drive

14" Color Monitor

$457

PLUS FREE $49.95 Oil Barons Adventure Program

C128 COMMODORE COMPUTER

NEW

(Order Now)

* $229^05

(SEE BELOW)

With $59.95 Timeworks Wordwriter Wordprocessor savings applied

- **340K 1571 Disk Drive $259.00**
- **Voice Synthesizer $39.95**
- **12" Monitor $79.95**

PRICES MAY BE LOWER

COMMODORE 64 COMPUTER $139.95

You pay only $139.95 when you order the powerful 84K COMMODORE 64 COMPUTER! LESS the value of the SPECIAL SOFTWARE DISCOUNT COUPON we pack with your computer that allows you to SAVE OVER $250 off software sale prices!! With only $100 of savings applied, your net computer cost is $39.95!!

* C128 DOUBLE SIDED DISKS 79¢ EA.

Get these 5¼" Double Sided Floppy Disks specially designed for the Commodore 128 Computer (1571 Disk Drive). 100% Certified, ***Lifetime Warranty***, Automatic Lint Cleaning Liner included. 1 Box of 10 - $9.90 (99¢ ea.), 5 Boxes of 10 - $44.50 (89¢ ea.), 10 Boxes of 10 - $79.00 (79¢ ea.).

13" COLOR MONITOR $149.95

You pay only $149.95 when you order this 13" COLOR MONITOR. LESS the value of the SPECIAL SOFTWARE DISCOUNT COUPON we pack with your monitor that allows you to save over $250 off software sale prices!! With only $100 of savings applied, your net color monitor cost is only $49.95. (16 Colors).

Premium Quality 120-140 CPS Comstar 10X Printer $148.00

The COMSTAR 10X gives you a 10" carriage, 120-140 CPS, 9 x 9 dot matrix with double strike capability for 18 x 18 dot matrix (near letter quality), high resolution bit image (120 x 144 dot matrix), underlining, back spacing, left and right margin setting, true lower decenders with super and subscripts, prints standard, italic, block graphics and special characters. It gives you print quality and features found on printers costing twice as much!! (Centronics Parallel Interface) List $399.00 **Sale $148.00.**

4 SLOT EXPANDER & 80 COLUMN BOARD $49.95

Now you program 80 COLUMNS on the screen at one time! Converts your Commodore 64 to 80 COLUMNS when you plug in the 80 COLUMN EXPANSION BOARD!! PLUS 4 slot expander! ***Limited Quantities. Sale $49.95.*** *Coupon $39.95*

80 COLUMNS IN COLOR PAPERBOCK WRITER 64 WORD PROCESSOR $39.95

This PAPERBOCK WRITER 64 WORD PROCESSOR is the finest available for the COMMODORE 64 computer! The ULTIMATE FOR PROFESSIONAL Word Processing, DISPLAYS 40 or 80 COLUMNS IN COLOR or black and white! Simple to operate, powerful text editing, complete cursor and insert/delete key controls line and paragraph insertion, automatic deletion, centering, margin settings and output to all printers! List $99.00. **SALE $39.95.** *Coupon $29.95.*

SPECIAL SOFTWARE COUPON

We pack a SPECIAL SOFTWARE DISCOUNT COUPON with every COMMODORE 64 COMPUTER, DISK DRIVE, PRINTER, or MONITOR we sell! This coupon allows you to SAVE OVER $250 OFF SALE PRICES!!

(Examples)

PROFESSIONAL SOFTWARE COMMODORE 64

Name	List	Sale	Coupon
PaperClip	$89.95	$34.95	$29.95
Consultant	$99.95	$49.95	$39.95
Leader Board	$39.95	$24.95	$22.95
The Print Shop	$44.95	$27.95	$26.95
Halley's Project	$39.95	$22.95	$19.95
Practicalc (spread sheet)	$59.95	$19.95	$14.95
Voice Command Module	$79.95	$39.95	$34.95
Nine Princes in Amber	$32.95	$24.95	$21.95
Super Bowl Sunday	$35.00	$22.95	$19.95
Flip and File Disk Filer	$24.95	$14.95	$12.95
Pro Joy Stick	$19.95	$12.95	$10.00
PartyWare	$19.95	$14.95	$11.95
Dust Cover	$ 8.95	$ 6.95	$ 4.60
Financial Planner Sylvia Porter	$59.95	$38.95	$35.95
Hardball	$29.95	$18.95	$16.95
C64 Troubleshoot & Repair Guide	$24.95	$15.95	$12.95

(See over 100 coupon items in our catalog)

Write or call for Sample SPECIAL SOFTWARE COUPON!

ATTENTION Computer Clubs

We Offer Big Volume Discounts

CALL TODAY!

PROTECTO WARRANTY

All Protecto's products carry a minimum 90 day warranty. If anything fails within 90 days from the date of purchase, simply send your product to us via United Parcel Service prepaid. We will IMMEDIATELY send you a replacement at no charge via United Parcel Service prepaid. This warranty proves once again that ***We Love Our Customers.***

* C128 COMMODORE COMPUTER $289.00

You pay only $289.00 for the C128 computer and we include the C128 Wordwriter Wordprocessor by Timeworks (Sale $59.95). Thus, your net cost for the C128 computer is only $229.05. List $349.00. **SALE $289.00.**

340K 1571 COMMODORE DISK DRIVE $259.00

Double Sided, Single Disk Drive for C-128 allows you to use C-128 mode plus CPM mode. 17 times faster than 1541, plus runs all 1541 formats. List $349.00. **Sale $259.00.**

SUPER AUTO DIAL MODEM $29.95

Easy to use. Just plug into your Commodore 64 computer and you're ready to transmit and receive messages. Easier to use than dialing your telephone, just push one key on your computer! Includes exclusive easy to use program for up and down loading to printer and disk drives. ***Best in U.S.A.*** List $99.00. **SALE $29.95.** *Coupon $24.95.*

VOICE SYNTHESIZER $39.95

For Commodore-64 computers. Just plug it in and you can program words and sentences, adjust volume and pitch, make talking adventure games, sound action games and customized talkies!! PLUS ($19.95 value) TEXT TO SPEECH program included FREE, just type a word and hear your computer talk — ADD SOUND TO "ZORK", SCOTT ADAMS AND OTHER ADVENTURE GAMES!! (Disk or tape.) List $89.00. **SALE $39.95**

12" MAGNAVOX (NAP) 80 COLUMN MONITOR WITH SOUND $79.95

Super High Resolution green screen monitor. 80 columns x 24 lines, easy to read, plus speaker for audio sound included. Fantastic value. List $129.00. **Sale $79.95.** (C128 cable $19.95. C64, Atari cable $9.95)

PRINTER/TYPEWRITER COMBINATION $229.95

"JUKI" Superb letter quality, daisy wheel printer/typewriter combination. Two machines in one — just a flick of the switch. 12" extra large carriage, typewriter keyboard, automatic margin control and relocate key, drop in cassette ribbon! (90 day warranty) centronics parallel or RS232 serial port built in (Specify). List $349.00. **SALE $229.95.** ***(Ltd. Qty.)***

14" RGB & COMPOSITE COLOR MONITOR $259.95

Must be used to get 80 columns in color with 80 column computers (C128 - IBM - Apple). (RGB Cable $19.95) Add $14.50 shipping. List $399.00. **SALE $259.95.**

- **LOWEST PRICES • 15 DAY FREE TRIAL**
- **BEST SERVICE IN U.S.A. • ONE DAY EXPRESS MAIL**

PHONE ORDERS
8 a.m. - 8 p.m. C.S.T. Weekdays
9 a.m. - 12 noon C.S.T. Saturdays

- **90 DAY FREE REPLACEMENT WARRANTY**
- **OVER 500 PROGRAMS • FREE CATALOGS**

Add $10.00 for shipping, handling and insurance. Illinois residents please add 6¼% tax. Add $20.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. Canadian orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES, EXCEPT CANADA. Enclose Cashier Check, Money Order or Personal Check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail! Prices & Availability subject to change without notice.
VISA — MASTER CARD — C.O.D. No. C.O.D. to Canada, APO-FPO

PROTECTO

We Love Our Customers

22292 N. Pepper Rd., Barrington, Illinois 60010

312/382-5244 to order

```
        {4 SPACES}{OFF}{WHT}-
        {BLK}E13 E3{GRN}{RVS}
        {SPACE}{OFF}"
SG 1310 PRINT"{RVS} {OFF}
        {21 SPACES}E*3{RVS}
        {2 SPACES}{OFF}{WHT}-
        {13 SPACES}{GRN}{RVS}
        {SPACE}{OFF}"
JD 1320 PRINT"{RVS} {OFF}
        {22 SPACES}{RVS}
        {2 SPACES}{OFF}{WHT}-
        {13 SPACES}{GRN}{RVS}
        {SPACE}{OFF}"
MD 1330 PRINT"{RVS} {OFF}
        {13 SPACES}{RVS}£
        {OFF}{7 SPACES}E*3
        {RVS} {OFF}{WHT}-
        {13 SPACES}{GRN}{RVS}
        {SPACE}{OFF}"
XQ 1340 PRINT"{RVS}{15 SPACES}
        {OFF}£{6 SPACES}{WHT}
        UC{GRN}{RVS} {OFF}
        {WHT}K{13 SPACES}{GRN}
        {RVS} {OFF}"
CD 1350 PRINT"{RVS}{4 SPACES}
        {OFF}£{17 SPACES}
        {WHT}-{GRN}{RVS}£
        {OFF}{7 SPACES}{RVS}
        {OFF}{6 SPACES}{RVS}
        {OFF}"
CH 1360 PRINT"{RVS}{3 SPACES}
        {OFF}£{17 SPACES}
        {RVS}£{3 SPACES}{OFF}
        {7 SPACES}{RVS} {OFF}
        {6 SPACES}{RVS} {OFF}"
DG 1370 PRINT"{RVS}{2 SPACES}
        {OFF}£{17 SPACES}
        {RVS}£{3 SPACES}{OFF}
        £{7 SPACES}{RVS}
        {OFF}{6 SPACES}{RVS}
        {OFF}"
DE 1380 PRINT"{RVS}{2 SPACES}
        {OFF}{8 SPACES}E53
        {RVS}£{GRN}
        {11 SPACES}{OFF}£
        {9 SPACES}{RVS} {OFF}
        {6 SPACES}{RVS} {OFF}
RE 1390 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}E53
        {RVS}£{4 SPACES}{GRN}
        {2 SPACES}{OFF}£
        {14 SPACES}{RVS}£
        {OFF}{6 SPACES}{RVS}
        {OFF}"
JS 1400 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {SPACE}E53{4 SPACES}
        {GRN} {OFF}£
        {14 SPACES}{RVS}£
        {2 SPACES}{OFF}
        {6 SPACES}{RVS} {OFF}"
HH 1410 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {SPACE}E53{2 SPACES}
        {GRN}{2 SPACES}{OFF}£
        {14 SPACES}{RVS}£
        {3 SPACES}{OFF}
        {6 SPACES}{RVS} {OFF}"
QA 1420 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {SPACE}E53{2 SPACES}
        {GRN} {OFF}£
        {14 SPACES}{RVS}£
        {2 SPACES}M {OFF}
        {6 SPACES}{RVS} {OFF}
BS 1430 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {3 SPACES}{OFF}£
        {14 SPACES}{RVS}£
        {2 SPACES}M {OFF}£
        {6 SPACES}{RVS} {OFF}"
RM 1440 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {2 SPACES}{OFF}£
        {14 SPACES}{RVS}£
        {2 SPACES}M {OFF}£
        {7 SPACES}{RVS} {OFF}"
RP 1450 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {SPACE}{OFF}£
        {14 SPACES}{RVS}£
        {2 SPACES}M {OFF}£
        {8 SPACES}{RVS} {OFF}"
CE 1460 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{7 SPACES}{RVS}
        {SPACE}{OFF}
        {14 SPACES}{RVS}£
        {4 SPACES}{OFF}£
        {9 SPACES}{RVS} {OFF}"
FJ 1470 PRINT"{RVS} {OFF}{WHT}
        -{GRN}{20 SPACES}E53
        {RVS}£ {GRN}
        {5 SPACES}{OFF}
        {10 SPACES}{RVS} {OFF}
        "
KX 1480 PRINT"{RVS} {OFF}{WHT}
        JI{GRN}{18 SPACES}E53
        {RVS}£ {GRN}
        {3 SPACES}{OFF}{WHT}-
        {SPACE}{GRN}{RVS}
        {OFF}{10 SPACES}{RVS}
        {SPACE}{OFF}"
SH 1490 PRINT"{RVS} E53
        {3 SPACES}{OFF}
        {17 SPACES}{RVS} {GRN}
        {3 SPACES}{OFF} {WHT}
        EQ3C{GRN}{RVS} {OFF}
        {10 SPACES}{RVS} {OFF}
        "
MX 1500 PRINT"{RVS} E53
        {3 SPACES}{OFF}
        {16 SPACES}{GRN}{RVS}
        £{4 SPACES}{OFF}{WHT}
        CEW3 {GRN}{RVS} {OFF}
        {10 SPACES}{RVS} {OFF}
        "
FF 1510 PRINT"{RVS} E53
        {4 SPACES}{OFF}{WHT}CC
        CI{10 SPACES}{GRN}
        {RVS}£{2 SPACES}{OFF}
        {WHT} - UEW3 {GRN}
        {RVS} {OFF}{10 SPACES}
        {RVS} {OFF}"
QR 1520 PRINT"{RVS} E53
        {7 SPACES}{OFF}{WHT}-
        {8 SPACES}{GRN}ED3
        {RVS}{4 SPACES}{OFF}
        {WHT}CK {RVS} {OFF}JC
        {GRN}{RVS} {OFF}{WHT}C
        CCCCCCCCC{GRN}{RVS}
        {OFF}"
KX 1530 PRINT"{RVS}{39 SPACES}
        {OFF}";:POKE2023,224:P
        OKE56295,5
DA 1540 RETURN
CS 1550 X=110:Y=165:COLOR4,16
HK 1560 PRINT"{CLR}E53{RVS}
        {17 SPACES}CCCCCCCC
        {3 SPACES}{OFF}E11 U3
        {RVS}EF3{OFF}"
SD 1570 PRINT"EJ3{6 SPACES}EL3
        EH3{15 SPACES}E*3{RVS}
        {3 SPACES}{OFF}
        {11 SPACES}EL3"
AD 1580 PRINT"EJ3{6 SPACES}
        {WHT}EC3EV3E53
        {16 SPACES}E*3{RVS} -
        {OFF}{11 SPACES}EL3
KB 1590 PRINT"EJ3{25 SPACES}
        E*3{RVS}-{OFF}
        {11 SPACES}EL3"
HK 1600 PRINT"EJ3{26 SPACES}
        {RVS}-{OFF}{11 SPACES}
        EL3"
AE 1610 PRINT"{RVS}{16 SPACES}
        {OFF}EF3{22 SPACES}EL3
        "
EQ 1620 PRINT"{RVS}{17 SPACES}
        {OFF}EF3{15 SPACES}
        {RVS}-{OFF}{5 SPACES}
        EL3"
XF 1630 PRINT"{BLK}E6 E3+E6 E3
        +E2 E3E53{RVS}
        {3 SPACES}{OFF}EF3
        {12 SPACES}ED3{RVS}
        {OFF}{5 SPACES}EL3"
HA 1640 PRINT"EJ3{5 SPACES}
        {WHT}{RVS}EI3{OFF}
        {6 SPACES}{RVS}EI3
        {OFF}{2 SPACES}E53
        {RVS}{17 SPACES}ED3
        {OFF}{5 SPACES}EL3"
MJ 1650 PRINT"EJ3{15 SPACES}
        {RVS}{3 SPACES}{OFF}
        {WHT}+++EW3{9 SPACES}
        E53{RVS}EF3EC3{OFF}
        {5 SPACES}EL3"
XA 1660 PRINT"EJ3{15 SPACES}
        {WHT}E3 T3 EW3EZ3
        {10 SPACES}E53{RVS}EV3
        {OFF}{WHT}UI
        {3 SPACES}E53EL3"
GQ 1670 PRINT"EJ3{4 SPACES}
        {RED}E9 @3E53
        {18 SPACES}{RVS} {OFF}
        {BLU}ED3{RVS} {OFF}
        {WHT}-{3 SPACES}E53EL3
        EJ3{4 SPACES}E33{RVS}
        {9 SPACES}{OFF}{WHT}
        EO3{17 SPACES}E53{RVS}
        {3 SPACES}{OFF}{WHT}-
        E53{3 SPACES}EL3EJ3
        {4 SPACES}E33{RVS} EC3
        EV3EC3EV3EC3EV3EC3EV3
        {OFF}{21 SPACES}{WHT}-
        E53{3 SPACES}EL3"
KJ 1680 PRINT"EJ3{5 SPACES}
        {RVS} {OFF}E73-E53
        {RVS} {OFF}{5 SPACES}
        E33{RVS} C C {OFF}
        {WHT}EU3{15 SPACES}-
        {3 SPACES}E53EL3"
DX 1690 PRINT"EJ3{5 SPACES}
        {RVS} {OFF}E73-E53
        {RVS} {OFF}{5 SPACES}
        {RVS}{14 SPACES}E*3
        {OFF}{6 SPACES}{WHT}-
        {3 SPACES}E53EL3"
SK 1700 PRINT"EJ3{5 SPACES}
        {RVS} {OFF}E73-E53
        {RVS} {OFF}{18 SPACES}
        E*3{RVS} E*3{OFF}
        {5 SPACES}{WHT}-
        {3 SPACES}E53EL3"
SJ 1710 PRINT"EJ3{5 SPACES}
        {RVS} {OFF}E73-E53
        {RVS} {OFF}{19 SPACES}
        E*3{RVS} E*3{OFF}
        {4 SPACES}{WHT}-
        {3 SPACES}E53EL3EJ3
        {5 SPACES}{RVS} E73
        E53 {OFF}{20 SPACES}
        E*3{RVS} {OFF}
        {4 SPACES}{WHT}-
        {3 SPACES}E53EL3"
KM 1720 PRINT"EJ3{5 SPACES}
        {RVS} {OFF} {RVS}
        {OFF}{21 SPACES}{RVS}
        {SPACE}{OFF}{4 SPACES}
        {BLK}EU3{3 SPACES}E53
        EL3"
BG 1730 PRINT"EJ3{5 SPACES}
        {RVS}EF3{2 SPACES}EC3
        {OFF}{20 SPACES}{RVS}
        {SPACE}{OFF}{8 SPACES}
        EL3"
FK 1740 PRINT"EJ3{6 SPACES}
        {RVS}EF3{2 SPACES}
        E11 P3 {OFF}{8 SPACES}
        {RVS} {OFF}{8 SPACES}
        EL3"
XQ 1750 PRINT"EJ3{29 SPACES}
        {RVS} {OFF}{8 SPACES}
        EL3"
```

From the publishers of *COMPUTE!*

June 1986 *COMPUTE!* Disk

All the exciting programs from the past three issues of *COMPUTE!* are on one timesaving, error-free floppy disk that is ready to load on your Atari 400/800, XL, and XE. The June 1986 *COMPUTE! Disk* contains the entertaining and useful Atari programs from the April, May, and June 1986 issues of *COMPUTE!*.

The June 1986 *COMPUTE! Disk* costs $12.95 plus $2.00 shipping and handling and is available only from COMPUTE! Publications.

For added savings and convenience, you may also subscribe to the *COMPUTE! Disk*. At a cost of only $39.95 a year (a $12.00 savings), you'll receive four disks, one every three months. Each disk will contain all the programs for your Commodore machine from the previous three issues of *COMPUTE!*.

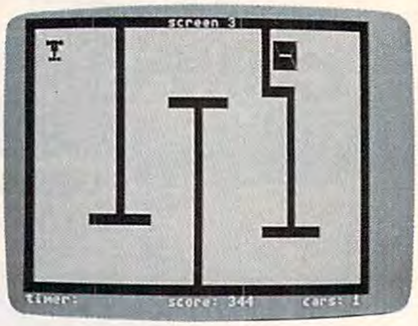
This is an excellent way to build your software library while you enjoy the quality programs from *COMPUTE!*.

Disks and subscriptions are available for Apple, Atari, Commodore 64 and 128, and IBM personal computers. Call for details.

For more information or to order the June 1986 *COMPUTE! Disk*, call toll free 1-800-346-6767 (in NY 212-265-8360) or write *COMPUTE! Disk*, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications.

```
CP 1760 PRINT"{J}{29 SPACES}
        {RVS} {OFF}{8 SPACES}
        {L}"
PX 1770 PRINT"{RVS} {29 U}
        {9 SPACES}{OFF}";:POKE
        2023,224:POKE56295,12
CA 1780 RETURN
AQ 1800 I=3456
JS 1810 READ A:IF A=256 THEN R
        ETURN
JB 1820 POKE I,A:I=I+1:GOTO181
        0
HP 1830 DATA000,000,000,000,00
        0,000,000,000
KK 1840 DATA000,004,000,000,00
        9,000,000,040
DX 1850 DATA000,000,106,000,00
        0,018,128,016
KE 1860 DATA000,162,020,000,04
        3,133,000,011
PB 1870 DATA225,000,010,232,00
        0,018,168,000
EA 1880 DATA020,160,000,005,12
        8,000,001,000
FS 1890 DATA000,000,000,000,00
        0,000,000,000
EX 1900 DATA000,000,000,000,00
        0,000,000,000
DS 1910 DATA000,000,000,000,00
        0,000,000,000
SB 1920 DATA000,000,000,000,00
        0,000,000,000
AA 1930 DATA000,000,000,000,08
        4,000,000,084
MC 1940 DATA064,000,016,064,01
        0,168,170,171
MS 1950 DATA232,170,171,232,06
        4,010,168,064
GP 1960 DATA000,016,000,000,08
        4,000,000,084
BF 1970 DATA000,000,000,000,00
        0,000,000,000
QE 1980 DATA000,000,000,000,00
        0,000,000,000
PH 1990 DATA000,000,000,000,00
        0,000,000,000
KF 2000 DATA000,000,000,000,00
        0,000,000,000
AM 2010 DATA000,000,000,001,00
        0,000,005,128
JB 2020 DATA000,020,160,000,01
        8,232,000,011
ED 2030 DATA232,000,011,161,00
        0,042,133,000
QD 2040 DATA162,020,018,128,01
        6,106,000,000
XB 2050 DATA040,000,000,009,00
        0,000,004,000
EK 2060 DATA000,000,000,000,00
        0,000,000,000
BJ 2070 DATA000,000,000,000,00
        0,000,000,000
CP 2080 DATA000,000,000,000,00
        0,000,000,000
QP 2090 DATA000,000,004,168,06
        4,005,169,064
XG 2100 DATA005,169,064,004,18
        4,064,000,184
FH 2110 DATA000,000,184,000,00
        0,168,000,000
CG 2120 DATA168,000,000,032,00
        0,000,032,000
XP 2130 DATA000,032,000,000,03
        2,000,001,033
XX 2140 DATA000,001,169,000,00
        1,033,000,000
RX 2150 DATA000,000,000,000,00
        0,000,000,000
SS 2160 DATA000,000,000,000,00
        0,000,000,000
DD 2170 DATA000,000,000,064,00
        0,002,080,000
XJ 2180 DATA010,020,000,043,13
        2,000,043,224
JH 2190 DATA000,074,224,000,08
        2,168,000,020
EB 2200 DATA138,000,004,002,13
        2,000,000,169
FD 2210 DATA000,000,040,000,00
        0,096,000,000
JG 2220 DATA016,000,000,000,00
        0,000,000,000
KC 2230 DATA000,000,000,000,00
        0,000,000,000
MF 2240 DATA000,000,000,000,00
        0,000,000,000
EQ 2250 DATA000,000,021,000,00
        0,021,000,000
CC 2260 DATA004,000,001,042,16
        0,001,043,234
DJ 2270 DATA170,043,234,170,04
        2,160,001,004
SA 2280 DATA000,001,021,000,00
        0,021,000,000
MK 2290 DATA000,000,000,000,00
        0,000,000,000
JG 2300 DATA000,000,000,000,00
        0,000,000,000
FG 2310 DATA000,000,000,000,00
        0,000,000,000
RC 2320 DATA000,000,000,016,00
        0,000,096,000
SD 2330 DATA000,040,000,000,16
        9,004,002,132
CJ 2340 DATA020,138,000,082,16
        8,000,075,224
HF 2350 DATA000,047,160,000,04
        2,132,000,010
XB 2360 DATA020,000,002,080,00
        0,000,064,000
GR 2370 DATA000,000,000,000,00
        0,000,000,000
BQ 2380 DATA000,000,000,000,00
        0,000,000,000
GD 2390 DATA000,000,000,001,03
        3,000,001,169
CX 2400 DATA000,001,033,000,00
        0,032,000,000
KC 2410 DATA032,000,000,032,00
        0,000,032,000
HM 2420 DATA000,032,000,000,16
        8,000,000,184
QR 2430 DATA000,000,184,000,00
        4,184,064,005
HD 2440 DATA169,064,005,169,06
        4,004,168,064
CA 2450 DATA000,000,000,000,00
        0,000,000,000
RD 2460 DATA000,000,000,000,00
        0,000,000,000
JC 2470 DATA000,000,000,000,00
        0,000,000,000
HG 2480 DATA000,000,000,000,00
        0,020,000,000
SP 2490 DATA085,000,001,085,06
        4,005,085,080
QS 2500 DATA021,085,084,085,08
        5,085,106,150
JB 2510 DATA169,101,150,089,10
        1,150,089,106
AB 2520 DATA150,169,106,150,16
        9,106,150,169
BA 2530 DATA106,150,169,106,15
        0,169,106,150
JQ 2540 DATA169,106,150,169,10
        6,150,169,000
RB 2550 DATA000,000,000,000,00
        1,000,064,007
ED 2560 DATA064,064,007,064,00
        0,049,000,004
JB 2570 DATA000,010,032,000,01
        4,000,128,000
PJ 2580 DATA002,136,192,066,04
        2,131,000,168
MS 2590 DATA000,048,043,032,00
        2,168,012,016
BM 2600 DATA032,128,000,000,04
        0,003,000,064
BH 2610 DATA048,000,000,000,01
        6,016,131,000
QX 2620 DATA116,160,000,116,00
        0,000,016,000,256
```



Smash-ups are commonplace in the Commodore 64 version of "Miami Ice."

Program 2: Miami Ice For Commodore 64

Version by Kevin Mykytyn, Editorial Programmer

Please refer to the "MLX" article in this issue before entering the following listing.

```
0801:0C 08 0A 00 9E 20 32 30 64
0809:36 32 00 00 00 20 E0 0E 11
0811:20 BC 0D A9 00 A0 18 B9 09
0819:A2 08 99 00 D4 88 10 F7 90
0821:20 C4 08 20 75 0A AD 01 A5
0829:DC 29 10 F0 F9 A2 00 A0 FD
0831:10 18 20 F0 FF A9 E2 A0 6F
0839:0B 20 1E AB A6 B4 E8 A9 D8
0841:00 20 CD BD A9 C8 85 F8 63
0849:A9 00 85 F9 20 D6 0D 20 15
0851:52 0A 20 95 0E A2 64 88 B6
0859:D0 FD CA D0 FA AD 1F D0 55
0861:AD 1E D0 AD 01 DC 29 10 A2
0869:D0 E8 A9 40 8D 04 D4 A9 25
0871:41 8D 04 D4 20 1D 09 20 FA
0879:48 09 CE 21 13 D0 09 AD 77
0881:22 13 8D 21 13 20 95 0E 7D
0889:CE 25 13 D0 0C AD 26 13 30
0891:8D 25 13 20 16 0E 20 4F 8E
0899:0D A0 00 88 D0 FD 4C 75 6D
08A1:08 00 05 00 01 00 19 F0 81
08A9:00 1E 00 00 00 89 00 00 67
08B1:0A 00 00 00 2B 00 00 05 25
08B9:F1 4F 20 CD BD A9 20 20 6C
08C1:D2 FF 60 A9 00 85 C3 85 05
08C9:C4 A9 00 85 B4 A9 03 85 D6
08D1:BD A9 93 20 D2 FF A2 03 7E
08D9:8E 21 D0 E8 8E 20 D0 A2 5B
08E1:07 A0 0B 18 20 F0 FF A9 EE
08E9:EC A0 0B 20 1E AB A5 B4 9B
08F1:18 69 31 8D A7 05 A9 0A 16
08F9:20 45 0A AD 01 DC 4A B0 48
0901:0A A5 B4 C9 06 F0 E7 E6 5F
0909:B4 10 E3 4A B0 08 A5 B4 40
0911:F0 DC C6 B4 10 D8 4A 4A B9
0919:4A B0 D3 60 C6 F7 D0 26 DA
0921:A9 C8 85 F7 A5 F8 05 F9 7F
0929:F0 1C A5 F8 38 E9 01 85 EF
0931:F8 A5 F9 E9 00 85 F9 A2 B3
0939:18 A0 07 18 20 F0 FF A6 4D
0941:F8 A5 F9 20 BB 08 60 AD E6
0949:23 13 10 05 49 FF 18 69 E7
0951:01 85 02 AD 24 13 10 05 F2
0959:49 FF 18 69 01 18 65 02 DE
0961:8D 00 D4 AD 1F D0 29 08 46
0969:D0 14 AD 1E D0 85 02 29 4A
0971:0A C9 0A F0 09 A5 02 29 57
0979:0C C9 0C F0 3A 60 A9 40 7B
```

```
0981:8D 04 D4 A9 80 8D 12 D4 C3
0989:A9 81 8D 12 D4 A9 08 8D 8E
0991:20 13 A9 05 20 45 0A AD D5
0999:20 13 C9 0D F0 06 EE 20 28
09A1:13 4C 93 09 A9 07 8D 15 EC
09A9:D0 A9 64 20 45 0A C6 BD BA
09B1:F0 5A 68 68 4C 4D 08 A9 B7
09B9:40 8D 04 D4 A9 00 85 02 77
09C1:A5 F8 05 F9 F0 32 A5 F8 B9
09C9:38 E9 01 85 F8 A5 F9 E9 27
09D1:00 85 F9 A5 B4 85 03 E6 87
09D9:C3 D0 02 E6 C4 20 52 0A 06
09E1:E6 02 A5 02 8D 01 D4 A9 80
09E9:40 8D 04 D4 A9 41 8D 04 BE
09F1:D4 C6 03 10 E2 4C C1 09 56
09F9:A9 40 8D 04 D4 E6 B4 A9 38
0A01:64 20 45 0A A9 05 8D 01 16
0A09:D4 4C 24 08 20 52 0A A9 A7
0A11:00 8D 15 D0 A2 0C A0 05 C3
0A19:18 20 F0 FF A9 9E A0 0B 73
0A21:20 1E AB A2 0B 20 38 0A BF
0A29:A2 0D 20 38 0A AD 01 DC 3F
0A31:29 10 D0 F9 4C 21 08 A0 2F
0A39:05 18 20 F0 FF A9 C1 A0 B3
0A41:0B 4C 1E AB 85 02 A9 00 F3
0A49:85 A2 A5 A2 C5 02 D0 FA 7A
0A51:60 A9 9F 20 D2 FF A2 18 E9
0A59:A0 17 18 20 F0 FF A6 C3 21
0A61:A5 C4 20 BB 08 A2 18 A0 D4
0A69:26 18 20 F0 FF A6 BD A9 69
0A71:00 4C CD BD A9 93 20 D2 DC
0A79:FF A9 01 8D 21 D0 A9 03 93
0A81:8D 20 D0 A2 18 A0 00 18 04
0A89:20 F0 FF A9 71 A0 0B 20 C8
0A91:1E AB A0 27 A9 A0 99 00 29
0A99:04 99 98 07 A9 04 99 00 2A
0AA1:D8 99 98 DB 88 10 ED A9 63
0AA9:00 85 FB 85 FD A9 04 85 1B
0AB1:FC A9 D8 85 FE A2 18 A0 75
0AB9:00 A9 A0 91 FB A0 27 91 A7
0AC1:FB A9 04 91 FD A0 00 91 DB
0AC9:FD A5 FB 18 69 28 85 FB 3A
0AD1:A5 FC 69 00 85 FC A5 FD 8E
0AD9:18 69 28 85 FD A5 FE 69 9F
0AE1:00 85 FE CA D0 D1 A9 00 05
0AE9:85 FB A9 04 85 FC A6 B4 57
0AF1:E0 07 90 02 A2 06 BD 09 1C
0AF9:0D 0A AA BD FB 0C 85 03 66
0B01:BD FC 0C 85 04 A0 FF C8 7A
0B09:B1 03 D0 19 C8 B1 03 8D 05
0B11:02 D0 8D 04 D0 8D 00 D0 DB
0B19:C8 B1 03 8D 03 D0 8D 05 B4
0B21:D0 8D 01 D0 60 84 02 48 91
0B29:48 29 1F 8D 28 13 68 4A 13
0B31:4A 4A 4A 29 06 AA A0 00 F6
0B39:68 10 02 A0 A0 8C 29 13 6E
0B41:A5 FB 18 7D 37 0C 85 FB F4
0B49:A5 FC 7D 38 0C 85 FC A0 B5
0B51:00 AD 29 13 F0 11 91 FB 14
0B59:A5 FB 85 FD A5 FC 18 69 8C
0B61:D4 85 FE A9 04 91 FD A4 C4
0B69:02 CE 28 13 D0 D2 F0 97 B5
0B71:12 9F 54 49 4D 45 52 3A F5
0B79:20 20 20 20 20 20 20 20 8F
0B81:20 20 53 43 4F 52 45 3A D6
0B89:20 20 20 20 20 20 20 20 9F
0B91:20 20 43 41 52 53 3A 20 B0
0B99:20 9D 94 20 00 12 9F 20 63
0BA1:92 47 41 4D 45 20 4F 56 6F
0BA9:45 52 20 2D 20 50 52 45 F9
0BB1:53 53 20 46 49 52 45 42 0F
0BB9:55 54 54 4F 4E 12 20 00 0A
0BC1:12 9F 20 20 20 20 20 20 B0
0BC9:20 20 20 20 20 20 20 20 DF
0BD1:20 20 20 20 20 20 20 20 E7
0BD9:20 20 20 20 20 20 20 20 EF
0BE1:00 9C 12 53 43 52 45 45 C9
0BE9:4E 20 00 92 9C 4D 20 49 FB
0BF1:20 41 20 4D 20 49 20 20 C7
0BF9:20 49 20 43 20 45 11 11 F3
0C01:11 9D 9D 9D 9D 9D 9D 9D D2
0C09:9D 9D 9D 9D 9D 1F 53 43 38
0C11:52 45 45 4E 20 31 11 11 2A
0C19:11 9D 9D 9D 9D 9D 9D 9D EA
0C21:9D 9D 9D 9D 9D 50 52 45 15
0C29:53 53 20 46 49 52 45 42 88
0C31:55 54 54 4F 4E 00 D8 FF AC
0C39:FF FF 28 00 01 00 4B FF F4
0C41:86 CD 00 37 B4 6A D1 A3 1A
0C49:E6 66 C6 90 A3 E6 65 86 5A
0C51:C5 E3 CC A3 E6 00 FB 42 8A
0C59:45 EC 47 25 EE 8B 6A 43 48
0C61:CD AA C2 02 B4 45 65 00 09
0C69:2D C8 68 C3 45 A7 EB C3 F7
0C71:83 E4 83 C3 E6 C3 83 E2 61
0C79:A1 04 83 47 E6 83 C3 E3 F8
0C81:C3 45 E5 AB 83 C6 83 AA 2D
0C89:83 C6 83 AA 83 C6 45 65 57
0C91:83 00 FA BC 66 C6 A1 E3 0C
0C99:A2 C8 E1 27 E2 A1 65 C4 11
0CA1:45 24 E1 63 05 EA C2 82 B3
0CA9:EA 81 C1 E5 46 81 05 E3 54
0CB1:06 E4 A1 C2 82 AD 81 C1 F5
0CB9:A6 C2 82 A4 84 E1 A1 82 E1
0CC1:A1 E1 81 68 05 C3 A1 E1 36
0CC9:C3 EA C1 81 E2 06 C1 00 81
0CD1:DC 46 65 D3 E9 64 E2 81 FB
0CD9:C1 EE 8E A2 24 AA 24 A4 43
0CE1:C9 E8 64 EC 00 73 89 46 9B
0CE9:F2 82 E1 C2 EC C5 E2 C1 8A
0CF1:A2 C4 AC C2 A1 82 B1 00 C8
0CF9:2D C8 3F 0C 46 0C 59 0C A4
0D01:6B 0C 95 0C D3 0C E8 0C F3
0D09:00 06 01 05 02 04 03 CE 0A
0D11:27 13 D0 39 A9 07 8D 27 DC
0D19:13 CE 2A 13 D0 0F A9 4B 48
0D21:8D 2A 13 AD 26 13 C9 07 E1
0D29:F0 03 EE 26 13 AD 23 13 65
0D31:F0 0B 10 06 EE 23 13 4C 5F
0D39:3E 0D CE 23 13 AD 24 13 6C
0D41:F0 0B 10 06 EE 24 13 4C 73
0D49:4E 0D CE 24 13 60 AE 20 81
0D51:13 AD 23 13 18 7D 77 0D A8
0D59:C9 46 90 04 C9 B9 90 03 95
0D61:8D 23 13 AD 24 13 18 7D 63
0D69:7F 0D C9 46 90 04 C9 B9 06
0D71:90 03 8D 24 13 60 01 00 A4
0D79:FF FF FF 00 01 01 FF FF 9F
0D81:FF 00 01 01 01 00 A9 01 28
0D89:8D 19 D0 AD 1B 13 8D 06 EB
0D91:D0 AD 1E 13 8D 07 D0 AD 4C
0D99:1C 13 0A 0A 0A 8D 10 D0 DF
0DA1:AD 20 13 18 69 D4 8D FB 34
0DA9:07 A9 FA 8D 12 D0 AD 0D 26
0DB1:DC 29 01 F0 03 4C 31 EA 4A
0DB9:4C BC FE A9 1B 8D 11 D0 A5
0DC1:A9 7F 8D 0D DC A9 87 8D 3D
0DC9:14 03 A9 0D 8D 15 03 A9 25
0DD1:81 8D 1A D0 60 A9 00 8D 97
0DD9:1C 13 8D 23 13 8D 24 13 D4
0DE1:A9 26 8D 1B 13 A9 3C 8D 03
0DE9:1E 13 A9 07 8D 20 13 A5 36
0DF1:A2 C5 A2 F0 FC A9 0F 8D 6C
0DF9:15 D0 A9 64 8D 21 13 8D F2
0E01:22 13 A9 07 8D 25 13 8D 4D
0E09:26 13 A9 07 8D 27 13 A9 7B
0E11:37 8D 2A 13 60 20 10 0D 53
0E19:AD 23 13 30 1A 18 6D 1A 60
0E21:13 8D 1A 13 AD 1B 13 69 08
0E29:00 8D 1B 13 AD 1C 13 69 AA
0E31:00 8D 1C 13 4C 57 0E 49 8A
0E39:FF 69 01 85 02 AD 1A 13 36
0E41:38 E5 02 8D 1A 13 AD 1B 9F
0E49:13 E9 00 8D 1B 13 AD 1C DE
0E51:13 E9 00 8D 1C 13 AD 24 F6
0E59:13 30 18 18 6D 1D 13 8D 23
0E61:1D 13 AD 1E 13 69 00 8D 34
0E69:1E 13 AD 1F 13 69 00 8D CC
0E71:1F 13 60 49 FF 18 69 01 B6
0E79:85 02 AD 1D 13 38 E5 02 A7
0E81:8D 1D 13 AD 1E 13 E9 00 F9
0E89:8D 1E 13 AD 1F 13 E9 00 4A
0E91:8D 1F 13 60 AD 01 DC 4A 1A
0E99:4A 4A B0 12 20 D3 0E EE FF
0EA1:20 13 AE 20 13 E0 08 D0 67
0EA9:05 A2 00 8E 20 13 4A B0 6C
0EB1:0D 20 D3 0E CE 20 13 10 E4
0EB9:05 A2 07 8E 20 13 4A B0 5D
0EC1:10 AD 26 13 C9 02 F0 09 88
0EC9:CE 26 13 AD 26 13 8D 25 D1
0ED1:13 60 48 A9 80 8D 0B D4 58
0ED9:A9 81 8D 0B D4 68 60 A0 37
0EE1:00 B9 5A 0F 99 00 35 B9 99
0EE9:5A 10 99 00 36 B9 5A 11 C8
0EF1:99 00 37 B9 5A 12 99 00 AB
0EF9:38 A9 00 99 00 39 88 D0 FC
0F01:E0 A0 3F B9 DA 12 99 80 0E
0F09:39 88 10 F7 A0 02 A9 FF C7
0F11:99 00 39 99 3C 39 88 10 A4
0F19:F7 A0 36 A9 80 99 03 39 66
0F21:88 88 88 10 F8 A9 0C 8D CB
0F29:5C 39 8D 62 39 A2 E4 8E 48
0F31:F9 07 E8 8E FA 07 E8 8E 68
0F39:FB 07 A0 03 B9 56 0F 99 B8
0F41:27 D0 88 10 F7 A9 08 8D 3D
0F49:1C D0 A9 00 8D 25 D0 A9 2B
0F51:07 8D 26 D0 60 02 00 02 35
0F59:02 00 00 00 00 00 00 00 78
0F61:00 00 00 00 10 00 00 60 60
0F69:00 00 28 00 00 A9 04 02 3D
0F71:84 14 8A 00 52 A8 00 4B A8
0F79:E0 00 2F A0 00 2A 84 00 A9
0F81:0A 14 00 02 50 00 00 40 8C
0F89:00 00 00 00 00 00 00 00 A7
0F91:00 00 00 00 00 00 00 00 AF
0F99:00 00 00 00 01 21 00 01 45
0FA1:A9 00 01 21 00 00 20 00 07
0FA9:00 20 00 00 20 00 00 20 F0
0FB1:00 00 20 00 00 A8 00 00 76
0FB9:B8 00 00 B8 00 04 B8 40 81
0FC1:05 A9 40 05 A9 40 04 A8 24
0FC9:40 00 00 00 00 00 00 00 08
0FD1:00 00 00 00 00 00 00 00 EF
0FD9:00 00 00 00 00 00 00 00 F7
0FE1:00 00 04 00 00 09 00 00 A4
0FE9:28 00 00 6A 00 00 12 80 67
0FF1:00 00 A2 14 00 2B 85 00 65
0FF9:0B E1 00 0A E8 00 12 A8 CA
1001:00 14 A0 00 05 80 00 01 65
1009:00 00 00 00 00 00 00 00 29
1011:00 00 00 00 00 00 00 00 31
1019:00 00 00 00 00 00 00 00 39
1021:00 00 00 00 00 00 00 00 41
1029:00 00 00 00 00 54 00 00 9A
1031:54 40 00 10 40 0A A8 AA B2
1039:AB E8 AA AB E8 40 0A A8 7E
1041:40 00 10 00 00 54 00 00 D4
1049:54 00 00 00 00 00 00 00 93
1051:00 00 00 00 00 00 00 00 71
1059:00 00 00 00 00 00 00 00 79
1061:00 00 00 00 00 00 00 00 81
1069:00 00 00 00 01 00 00 05 96
1071:80 00 14 A0 00 12 E8 00 78
1079:0B E8 00 0B A1 00 2A 85 F0
1081:00 A2 14 12 80 10 6A 00 07
1089:00 28 00 00 09 00 00 04 FF
1091:00 00 00 00 00 00 00 00 B1
1099:00 00 00 00 00 00 00 00 B9
10A1:00 00 00 00 00 00 00 00 C1
10A9:00 00 00 04 A8 40 05 A9 04
10B1:40 05 A9 40 04 B8 40 00 EF
10B9:B8 00 00 B8 00 00 A8 00 13
10C1:00 A8 00 00 20 00 00 20 2D
10C9:00 00 20 00 00 20 00 01 6F
10D1:21 00 01 A9 00 01 21 00 83
10D9:00 00 00 00 00 00 00 00 F9
10E1:00 00 00 00 00 00 00 00 02
10E9:00 00 00 00 40 00 02 50 60
10F1:00 0A 14 00 2B 84 00 2B AD
10F9:E0 00 4A E0 00 52 A8 00 7C
1101:14 8A 00 04 02 84 00 00 32
1109:A9 00 00 28 00 00 60 00 43
1111:00 10 00 00 00 00 00 00 37
1119:00 00 00 00 00 00 00 00 3B
1121:00 00 00 00 00 00 00 00 43
1129:00 00 00 15 00 00 15 00 C6
1131:00 04 00 01 2A A0 01 2B 65
1139:EA AA 2B EA AA 2A A0 01 CF
1141:04 00 01 15 00 00 15 00 01
1149:00 00 00 00 00 00 00 00 6B
1151:00 00 00 00 00 00 00 00 73
1159:00 00 00 00 00 00 00 00 7B
1161:01 00 00 07 40 10 07 40 05
1169:10 01 00 01 30 28 08 80 96
1171:38 02 80 C0 04 A2 03 00 FC
```

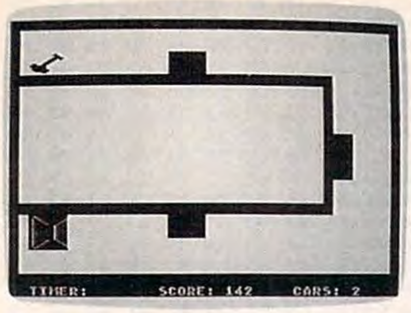
```
1179:98 01 0C AB 01 02 A8 80 46
1181:04 22 B0 03 00 60 0C 10 1E
1189:00 28 C0 40 20 01 D0 C0 39
1191:01 D0 00 00 40 00 00 00 6A
1199:00 00 00 00 00 01 00 40 FF
11A1:07 40 40 07 40 00 31 00 34
11A9:04 00 0A 20 00 0E 00 80 C9
11B1:00 02 88 C0 42 2A 83 00 33
11B9:A8 00 30 2B 20 02 A8 0C 4F
11C1:10 20 80 00 00 28 03 00 AA
11C9:40 30 00 00 00 10 10 83 FB
11D1:00 74 A0 00 74 00 00 10 D8
11D9:00 00 07 40 40 01 00 00 E6
11E1:00 00 04 30 00 00 00 00 87
11E9:80 00 00 02 80 00 00 8A FA
11F1:80 00 2A 0C 42 08 03 02 94
11F9:8B 00 C0 20 E0 CA 28 03 81
1201:D0 A8 00 02 00 28 30 20 F8
1209:02 0C 00 40 C0 00 00 00 3B
1211:10 01 03 00 07 80 00 07 1F
1219:00 00 01 00 10 30 00 00 9E
1221:00 00 00 00 00 02 88 00 5E
1229:02 20 00 00 32 A0 00 0A 74
1231:03 00 00 80 00 82 00 02 EA
1239:A8 00 20 08 E0 3A 00 00 26
1241:28 02 08 40 02 02 00 20 37
1249:00 00 00 00 30 00 10 00 0F
1251:00 00 00 10 00 03 00 01 83
1259:00 40 30 00 00 00 00 08 9B
1261:00 00 0A 00 00 00 08 00 D6
1269:00 30 2A 00 00 A0 00 00 61
1271:00 0A 00 20 02 00 80 00 2B
1279:08 A0 20 00 B8 CC 00 00 C6
1281:28 00 00 20 00 02 00 02 C5
1289:00 00 22 00 00 00 00 C0 B2
1291:00 00 00 00 01 00 10 00 DD
1299:00 08 30 00 28 00 00 00 07
12A1:00 00 00 02 00 00 0C 00 FD
12A9:00 00 0A A0 00 00 80 00 1A
12B1:00 00 00 02 00 00 08 00 06
12B9:00 0A 80 00 0B 30 00 20 A9
12C1:00 00 00 00 00 00 00 00 E5
12C9:00 00 00 00 00 00 00 00 ED
12D1:00 00 08 02 01 00 12 00 43
12D9:00 00 00 00 3F FF FC 5F 51
12E1:FF FA 5F FF FA 6F FF F6 3D
12E9:6F FF F6 77 FF EE 77 FF C6
12F1:EE 7B FF DE 7B F3 DE 7C 40
12F9:00 3E 7B F3 DE 7B FF DE 20
1301:77 FF EE 77 FF EE 6F FF D2
1309:F6 6F FF F6 5F FF FA 5F 46
1311:FF FA 3F FF FC 00 00 00 C5
1319:FF 00 00 00 00 00 00 00 3F
```

Program 3: Miami Ice For Atari

Version by Kevin Mykytyn, Editorial Programmer

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
GP 10 GOSUB 350:DIM B$(6),D$
      (1),DRAW$(1),GAR$(22)
FM 20 LV=1:POKE 82,0:GAR$(1,
      1)=CHR$(34):GAR$(2,22)
      ="#$%{DOWN}{4 LEFT}&'(
      ){DOWN}{4 LEFT}*+,-"
KE 30 GRAPHICS 17:POKE 559,6
      2:SETCOLOR 4,5,12:SETC
      OLOR 0,0,0:SETCOLOR 2,
      7,6:CARS=3:SCORE=0
PN 40 IF STRIG(0)=0 THEN 40
AF 50 POSITION 5,5:PRINT #6;
      "MIAMI ICE":POSITION 5
      ,9:PRINT #6;"SCREEN 1"
      :POSITION 2,13:PRINT #
      6;"PRESS FIREBUTTON"
CI 60 POSITION 12,9:PRINT #6
      ;CHR$(LV+176):FOR TD=1
       TO 200:NEXT TD
```

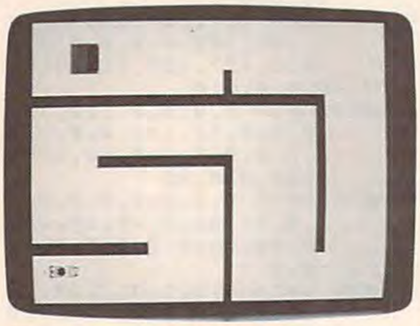


The sleek car in the Atari version of "Miami Ice" is composed of four player/missile graphics shapes.

```
GE 70 IF STICK(0)=14 AND LV<
      7 THEN LV=LV+1:GOTO 60
GC 80 IF STICK(0)=13 AND LV>
      1 THEN LV=LV-1:GOTO 60
JI 90 IF STRIG(0) THEN 70
BE 100 POKE 752,1:GOSUB 330:
       POKE 756,CHBAS:POKE 5
       4279,CHBAS:POKE 559,6
       2:POKE 53277,3
ID 110 POKE 709,74:POKE 710,
       0:POKE 708,0:POKE 712
       ,15:A=0:GOSUB 210
MP 120 FOR A=1 TO 21:POSITIO
       N 0,A:PRINT "!
       {38 SPACES}!":NEXT A:G
       OSUB 210
JM 130 GOSUB 310:GOSUB 400:T
       I=200
KK 140 GOSUB 320
DO 150 POKE 28,1:A=USR(29195
       ):A=USR(29184):POKE 5
       3278,255:POKE 209,0
PG 160 IF STRIG(0) THEN 160
JP 165 POKE 28,0:IF PEEK(209
       )>2 THEN 230
OF 170 ON PEEK(209) GOTO 230
       ,280:SOUND 1,70-(PEEK
       (29)-PEEK(30)),6,4:GO
       SUB 220:GOTO 165
ML 180 IF STICK(0)=11 THEN S
       P=SP+1-8*(SP=7)
JK 190 IF STICK(0)=7 THEN SP
       =SP-1+8*(SP=0)
BP 200 IF STRIG(0)=0 THEN V=
       V+(V<10)
CN 210 POSITION 0,A:PRINT "!
       !!!!!!!!!!!!!!!!!!!!!!
       !!!!!!!!!!!!!!!!!!!";:
       RETURN
DK 220 TI=TI-0.1*(TI>0):POSI
       TION 9,23:PRINT INT(T
       I);" ";:RETURN
LP 230 POKE 28,1:FOR A=8 TO
       11 STEP 0.5:POKE 205,
       A:SOUND 1,100,8,23-A:
       NEXT A:FOR A=11 TO 0
       STEP -0.1:SOUND 1,100
       ,8,A
JN 240 NEXT A:CARS=CARS-1:IF
        CARS<>0 THEN 140
DI 250 GOSUB 320:GOSUB 310:P
       OSITION 15,10:PRINT "
       GAME OVER":POSITION 1
       2,12:PRINT "PRESS FIR
       EBUTTON"
PI 260 IF STRIG(0) THEN 260
DF 270 GOTO 30
LE 280 POKE 28,1:IF TI THEN
       FOR A=1 TO INT(TI):SC
       ORE=SCORE+LV:GOSUB 30
       0:SOUND 1,200-A,10,10
       :NEXT A
HE 290 LV=LV+(LV<7):POKE 205
       ,11:SOUND 1,0,0,0:POK
       E 31,1:GOTO 130
BE 300 POSITION 22,23:PRINT
       SCORE;" ";:RETURN
JH 310 FOR A=1 TO 21:POSITIO
       N 1,A:PRINT "
       {38 SPACES}";:NEXT A:R
       ETURN
CP 320 POSITION 0,23:PRINT "
         TIMER:{7 SPACES}SCO
       RE:{8 SPACES}CARS: ";
       CARS;" ";:GOSUB 300:R
       ETURN
AH 330 GRAPHICS 0:POKE 559,6
       2:DL=PEEK(560)+256*PE
       EK(561):POKE DL+3,68:
       FOR I=DL+6 TO DL+27:P
       OKE I,4:NEXT I:POKE I
       ,2
CG 340 I=I+1:POKE I,65:POKE
       I+1,0:POKE I+2,DL/256
       :RETURN
IL 350 CHBAS=120:POKE 106,CH
       BAS-8:GRAPHICS 0:POKE
        752,1:POSITION 14,10
       :PRINT "PLEASE WAIT":
       CHSET=CHBAS*256
LF 360 GOSUB 660
MC 370 FOR A=0 TO 1023:POKE
       CHSET+A,PEEK(57344+A)
       :NEXT A:RESTORE 540:F
       OR A=CHSET+16 TO CHSE
       T+111:READ B
NK 380 POKE A,B:NEXT A:POKE
       756,CHBAS:FOR A=CHSET
       +8 TO CHSET+15:POKE A
       ,85:NEXT A:POKE 54279
       ,CHBAS:POKE 206,CHBAS
       +4
EK 390 POKE 559,62:POKE 623,
       4:POKE 704,55:POKE 70
       6,55:POKE 705,0:POKE
       707,0:POKE 53277,3:RE
       TURN
GK 400 SC=PEEK(88)+256*PEEK(
       89):RESTORE 440:RESTO
       RE 440+LV*10
JF 410 READ B$:IF B$="END" T
       HEN READ X,Y:POSITION
        X,Y:POKE 752,1:PRINT
        GAR$:RETURN
KJ 420 D$=B$(1,1):DRAW$=B$(2
       ,2):LENGTH=VAL(B$(3,L
       EN(B$)))
GG 430 FOR A=1 TO LENGTH:SC=
       SC-40*(D$="U")+40*(D$
       ="D")+(D$="R")-(D$="L
       "):IF DRAW$="Y" THEN
       POKE SC,1
CA 440 NEXT A:GOTO 410
BI 450 DATA DN11,RY33,UY7,DY
       14,END,2,17
JF 460 DATA DN5,RY16,UY2,RY3
       ,DY1,LY2,DY1,RY15,DY5
       ,RY2,DY3,LY1,UY2,LY1,
       DY5,LY13,DY2,LY3,UY1,
       RY2,UY1,LY17,END,2,17
DN 470 DATA RN10,DY18,LY4,RY
       7,UN12,RN2,RY7,LY3,DY
       16,RN5,UN4,RY10,LY3,U
       Y12,LY3,UY5,END,30,2
MA 480 DATA RN7,DY18,RY7,RN4
       ,RY16,UY14;LY12,LN4,L
       Y6,DY9,RY7,RN4,RY10,E
       ND,14,9
AD 490 DATA DN5,RY9,RN5,UN5,
       DY10,LY8,DN5,LN6,RY11
       ,DY3,UY3,RY8,UY10,DY1
       0,RY14,UY10,LN8,DN6,U
       Y10,END,2,16
IJ 500 DATA RN6,DY16,RY5,DY2
       ,UY2,RY6,DN7,UY3,UN4,
       RY5,DY2,UY2,RY5,DN7,U
       Y3,UN4,RY7,UN6,RN5,LY
```

```
16,UY5,DY5,LY10
MN 505 DATA UY5,RN5,UY4,RN11
,UN1,DY5,RY5,END,30,1
MK 510 DATA DN6,RY6,DY3,UY3,
RY4,UN6,DY2,DN4,RY8,U
Y2,DY2,RY1,DY3,UY3,RY
5,UN6,DY2,DN4,RY5,UY2
,DY2,RY4,DY3
AM 520 DATA DN5,RN6,LY9,LY4,
UY3,DY3,LY8,DY4,UY4,L
Y5,UY3,DY3,LY7,DY4,RN
6,UN1,DY5,RN13,DN1,UY
5,END,30,15
GP 540 DATA 85,122,110,107,1
06,106,106,106
IK 550 DATA 85,170,170,170,2
34,186,174,171
IC 560 DATA 85,170,170,170,1
70,171,174,250
IM 570 DATA 85,171,174,186,2
34,170,170,170
KD 580 DATA 106,106,106,106,
106,106,106,106
LE 590 DATA 171,171,171,171,
171,171,171,171
JM 600 DATA 250,250,250,250,
250,250,250,250
KF 610 DATA 170,170,170,170,
170,170,170,170
GO 620 DATA 106,106,106,106,
107,110,122,85
IJ 630 DATA 171,174,186,234,
170,170,170,85
IB 640 DATA 250,174,171,170,
170,170,170,85
IL 650 DATA 170,170,170,234,
186,174,171,85
ME 660 RESTORE 680:C=0:FOR A
=29184 TO 30512:READ
B:POKE A,B:C=C+B:NEXT
 A:IF C=93195 THEN RE
TURN
KG 670 PRINT "{CLEAR}ERROR I
N DATA":STOP
HN 680 DATA 160,87,162,114,1
69,7,32,92,228,104,96
,169,10,141,107,116,1
41,108,116,160,63,169
,0,153,47,119
GJ 690 DATA 136,16,250,169,5
,141,103,116,133,205,
169,55,141,98,116,141
,100,116,169,3,133,29
,133,30,169,0
DG 700 DATA 133,207,133,208,
133,203,169,7,141,109
,116,141,110,116,165,
206,133,204,162,3,160
,0,152,145,203,200
PG 710 DATA 208,251,230,204,
202,16,246,104,96,216
,169,0,133,77,32,126,
115,206,107,116,208,9
,173,108,116,141
BK 720 DATA 107,116,32,237,1
15,165,28,208,20,206,
109,116,208,9,173,110
,116,141,109,116,32,1
36,114,32,250,114
JK 730 DATA 32,65,116,76,98,
228,166,205,165,207,2
4,125,171,114,201,16,
144,4,201,241,144,2,1
33,207,165,208
CN 740 DATA 24,125,179,114,2
01,16,144,4,201,241,1
44,2,133,208,96,1,0,2
55,255,255,0,1,1,255,
255,255
HO 750 DATA 0,1,1,1,0,206,10
4,116,208,57,169,19,1
41,104,116,206,103,11
6,208,21,169,2,141,10
3,116,165
AH 760 DATA 29,201,3,240,2,1
98,29,165,30,201,3,24
0,2,198,30,165,207,24
0,9,16,5,230,207,76,2
36,114
PI 770 DATA 198,207,165,208,
240,9,16,5,230,208,76
,249,114,198,208,96,3
2,187,114,165,207,141
,105,116,166,29
NN 780 DATA 24,101,207,141,1
05,116,202,208,247,16
5,208,141,106,116,166
,30,24,101,208,141,10
6,116,202,208,247,173
CN 790 DATA 105,116,48,18,24
,109,97,116,141,97,11
6,173,98,116,105,0,14
1,98,116,76,78,115,73
,255,24,105
KC 800 DATA 1,141,101,116,17
3,97,116,56,237,101,1
16,141,97,116,173,98,
116,233,0,141,98,116,
173,106,116,48
FG 810 DATA 16,24,109,99,116
,141,99,116,173,100,1
16,105,0,141,100,116,
96,73,255,24,105,1,14
1,101,116,173
GA 820 DATA 99,116,56,237,10
1,116,141,99,116,173,
100,116,233,0,141,100
,116,96,173,98,116,14
1,0,208,141,1
LJ 830 DATA 208,24,105,8,141
,2,208,141,3,208,165,
205,133,203,169,0,133
,204,162,6,6,203,38,2
04,202,208
PG 840 DATA 249,165,203,24,1
05,111,141,198,115,16
5,204,105,116,141,199
,115,165,206,133,204,
169,4,141,102,116,169
BB 850 DATA 0,133,203,172,10
0,116,145,203,200,162
,0,189,255,255,145,20
3,200,232,224,16,208,
245,169,0,145,203
JP 860 DATA 173,198,115,24,1
05,16,141,198,115,173
,199,115,105,0,141,19
9,115,230,204,206,102
,116,208,205,96,165
DI 870 DATA 205,201,8,144,1,
96,173,120,2,74,74,74
,176,12,230,205,166,2
05,224,8,208,4,162,0,
134,205
II 880 DATA 74,176,8,198,205
,16,4,162,7,134,205,1
73,132,2,208,40,166,2
05,189,171,114,16,5,7
3,255,24
JM 890 DATA 105,1,24,101,29,
201,8,176,2,133,29,18
9,179,114,16,5,73,255
,24,105,1,24,101,30,2
01,8
HG 900 DATA 176,2,133,30,96,
165,209,208,27,162,0,
173,4,208,13,5,208,13
,6,208,13,7,208,74,14
4.2
KI 910 DATA 162,1,74,74,144,
2,162,2,134,209,96,0,
0,0,0,0,0,0,0,0,0,0
,0,0,0
AJ 920 DATA 0,0,0,0,0,0,1,3,
15,19,35,54,28,0,0,0,
0,0,0,0,0,0,0,48,112,
108
ME 930 DATA 28,9,3,3,0,0,0,8
,28,56,112,224,192,12
8,0,0,0,0,0,0,0,0,16,
48,0,6
NC 940 DATA 4,0,0,0,0,0,0,12
8,128,0,0,1,1,1,1,1,1
,1,1,1,1,1,3,2,2,3
MI 950 DATA 1,4,6,4,0,0,0,0,
0,0,0,0,4,5,5,4,0,128
,128,128,128,128,128,
128,128,128
IK 960 DATA 128,128,192,64,6
4,192,128,32,96,32,0,
0,0,0,0,0,0,0,32,160,
160,32,0,0,0,16
EL 970 DATA 56,28,14,7,3,1,0
,0,0,0,0,0,0,0,8,12,0
,96,32,0,0,0,0,0,0,1
JL 980 DATA 1,0,0,0,0,0,0,0,
0,0,128,192,240,200,1
96,108,56,0,0,0,0,0,0
,0,0,0
ED 990 DATA 0,12,14,54,56,14
4,192,192,0,0,0,0,0,0
,0,0,127,127,127,0,0,
0,0,0,0,0
NE 1000 DATA 0,0,0,0,112,32,
0,0,0,32,112,0,0,0,0
,0,0,0,0,0,0,60,230,
230,230,60
JG 1010 DATA 0,0,0,0,0,0,0,0
,0,56,56,0,24,24,24,
0,56,56,0,0,0,0,0,0,
0,0
KG 1020 DATA 0,0,1,3,7,14,28
,56,16,0,0,0,0,1,1,0
,0,0,0,0,0,32,96,0,1
2,8
NK 1030 DATA 0,0,0,56,108,19
6,200,240,192,128,0,
0,0,0,0,0,0,0,192,19
2,144,56,54,14,12,0
CK 1040 DATA 0,0,0,0,0,0,0,1
,3,2,2,3,1,1,1,1,1,1
,1,1,1,1,1,0,4,5
DC 1050 DATA 5,4,0,0,0,0,0,0
,0,0,4,6,4,128,192,6
4,64,192,128,128,128
,128,128,128,128,128
EK 1060 DATA 128,128,128,0,3
2,160,160,32,0,0,0,0
,0,0,0,0,32,96,32,0,
0,28,54,35,19,15
GE 1070 DATA 3,1,0,0,0,0,0,0
,0,0,3,3,9,28,108,11
2,48,0,0,0,0,0,0,0,0
,0
AJ 1080 DATA 0,0,0,0,0,0,128
,192,224,112,56,28,8
,0,0,0,0,128,128,0,0
,0,0,0,0,4
FL 1090 DATA 6,0,48,16,0,0,0
,0,0,0,0,60,103,103,
103,60,0,0,0,0,0,0,0
,0,0,28
JJ 1100 DATA 28,0,24,24,24,0
,28,28,0,0,0,0,0,0,0
,0,0,0,254,254,254,0
,0,0,0,0
LP 1110 DATA 0,0,0,0,0,0,14,
4,0,0,0,4,14,0,0,0,0
,0,1,5,0,3,8,11,0,1
JG 1120 DATA 4,10,5,1,4,1,0,
0,0,0,6,0,0,0,0,6,0,
0,0,6,0,0,0,0,0,64
EL 1130 DATA 128,64,160,64,1
60,160,64,128,32,64,
128,128,0,0,0,0,0,12
8,0,0,0,0,128,0,192,
0
GC 1140 DATA 0,0,0,2,0,4,17,
0,32,10,0,20,64,2,16
,1,4,0,0,0,0,0,8,0,0
,0
```

```
EE 1150 DATA Ø,Ø,Ø,Ø,Ø,8,Ø,Ø,Ø
        ,Ø,Ø,128,64,Ø,16,160
        ,8,160,48,Ø,136,32,Ø
        ,16,Ø,128,Ø
CL 1160 DATA Ø,4,Ø,Ø,Ø,Ø,Ø,Ø,Ø
        ,Ø,Ø,Ø,64,Ø,Ø,Ø,2,Ø,
        Ø,16,Ø,32,Ø,Ø,16,64,
        Ø
CB 1170 DATA Ø,1,Ø,32,Ø,Ø,Ø,
        Ø,8,Ø,Ø,Ø,Ø,Ø,Ø,Ø,16
        ,Ø,Ø,Ø,Ø,Ø,128,Ø,Ø,1
        6
ME 1180 DATA Ø,4,Ø,Ø,2,8,Ø,Ø
        ,16,Ø,128,Ø,Ø,Ø,Ø,Ø,
        Ø,Ø,Ø,Ø,Ø,Ø,Ø,Ø,4,Ø
MN 1190 DATA Ø,Ø,Ø
```



Watch out for those slick turns in the Apple version of "Miami Ice."

Program 4: Miami Ice For Apple

Version by Tim Victor, Editorial Programmer

Please refer to the "MLX" article in this issue before entering the following listing.

```
START ADDRESS: 1000
END ADDRESS:   1E97

1000: A9 00 85 EC A9 60 85 ED 3C
1008: A9 1A 85 FA A9 1A 85 FB A0
1010: 20 B3 17 20 78 17 A9 FF 85
1018: 8D C4 1E 20 97 18 A9 03 69
1020: 8D BB 1E A9 00 8D BF 1E 28
1028: 8D C0 1E F0 03 20 7E 19 C0
1030: 2C 57 C0 2C 52 C0 2C 54 59
1038: C0 2C 50 C0 20 B2 14 A9 77
1040: 00 A0 0A 91 FA A0 11 91 F0
1048: FA 2C 54 C0 A9 20 85 E6 47
1050: A9 40 8D 97 1E A9 01 8D A7
1058: 98 1E A9 01 8D 1D 1A A9 50
1060: 01 8D 1A 1A A9 01 8D 1B D0
1068: 1A A9 0A 8D 1C 1A A9 02 B8
1070: 8D 1E 1A A9 00 8D B1 1E 74
1078: A9 00 8D AF 1E 8D B0 1E C0
1080: 20 DA 12 AD 61 C0 30 FB EE
1088: 20 90 18 EE B9 1E D0 26 DC
1090: C9 78 90 15 C9 89 90 1E CA
1098: EE 1E 1A AD 1E 1A C9 08 CA
10A0: D0 11 A9 00 8D 1E 1A F0 AC
10A8: 0A CE 1E 1A 10 05 A9 07 D5
10B0: 8D 1E 1A 20 DA 12 AD 61 40
10B8: C0 10 CD 8D A5 1E AE 1E F0
10C0: 1A BD E3 12 8D A6 1E BD FB
10C8: EB 12 8D A7 1E A9 01 8D B6
10D0: A8 1E A9 00 8D B2 1E A9 1F
10D8: 00 8D C1 1E 8D C2 1E 20 4A
10E0: 90 18 C9 78 90 08 E9 10 98
10E8: C9 79 B0 02 A9 78 18 69 4B
10F0: 08 4A 4A 4A 4A 38 E9 08 A4
10F8: 8D A9 1E 48 AD A8 1E 69 48
1100: F8 2C A9 1E 30 0C 49 FF 05
1108: 38 69 00 CD A9 1E B0 0A AE
1110: 90 05 CD A9 1E 90 03 8D D6
1118: A9 1E 68 F0 19 38 ED A9 E1
1120: 1E F0 0A 30 04 A9 02 D0 6D
1128: 0D A9 FE D0 09 A9 01 2C 45
1130: A9 1E 10 02 A9 FF 8D AA E3
1138: 1E AD 61 C0 10 0F 2C A5 C7
1140: 1E 30 0A A0 08 CC A8 1E AB
1148: 90 03 EE A8 1E 8D A5 1E 6C
1150: AD A9 1E AC A8 1E 20 FD 3E
1158: 12 A5 51 48 AC A6 1E 20 F7
1160: F3 12 AD A7 1E 18 65 50 9D
1168: 8D A7 1E 68 AC A7 1E 20 E5
1170: F3 12 AD A6 1E 38 E5 50 1F
1178: 8D A6 1E AD A8 1E AC A6 67
1180: 1E 20 FD 12 A5 50 8D AB CF
1188: 1E A5 51 8D AC 1E AD A8 08
1190: 1E AC A7 1E 20 FD 12 A5 86
1198: 50 8D AD 1E A5 51 8D AE 1A
11A0: 1E AC B2 1E C8 D0 0C A0 77
11A8: 00 AD A8 1E C9 01 F0 03 64
11B0: CE A8 1E 8C B2 1E EE C1 9E
11B8: 1E D0 0A EE C2 1E D0 05 83
11C0: A9 FF 8D C2 1E 18 AD AF F1
11C8: 1E 6D AC 1E 8D AF 1E AD E1
11D0: 1B 1A 6D AB 1E C9 07 2C C1
11D8: AB 1E 30 09 90 0E EE 1A A3
11E0: 1A E9 07 B0 07 90 05 CE C9
11E8: 1A 1A 69 06 18 6D B1 1E 24
11F0: C9 07 90 05 E9 07 EE 1A 7F
11F8: 1A 8D 1B 1A 4D 1A 1A 29 C0
1200: 01 8D B1 1E F0 11 CE 1B A4
1208: 1A 10 0C AD 1B 1A 18 69 74
1210: 07 8D 1B 1A CE 1A 1A 18 4B
1218: AD B0 1E 6D AE 1E 8D B0 93
1220: 1E AD 1C 1A 6D AD 1E 8D CF
1228: 1C 1A AD A6 1E C9 20 90 E9
1230: 0A C9 E0 B0 06 C9 00 10 5A
1238: 0D 30 21 A0 00 2C A7 1E 3B
1240: 30 2E A0 04 D0 2A AD A7 8E
1248: 1E A0 02 C9 E0 B0 21 C9 56
1250: 20 90 1D C9 00 10 02 88 B5
1258: 24 C8 D0 14 AD A7 1E A0 05
1260: 06 C9 20 90 0B C9 E0 B0 F8
1268: 07 C9 00 10 02 C8 24 88 87
1270: 98 18 6D AA 1E 10 03 18 8E
1278: 69 08 C9 08 90 02 E9 08 75
1280: 8D 1E 1A 20 DA 12 AD B8 6B
1288: 1E F0 37 AD BA 1E F0 35 1F
1290: A9 20 38 ED C2 1E 90 1E 45
1298: 0A 0A 8D C5 1E AE B3 1E 83
12A0: AD C5 1E 18 6D BF 1E C9 C2
12A8: 64 90 05 EE C0 1E E9 64 69
12B0: 8D BF 1E CA 10 EA AD B3 37
12B8: 1E C9 06 F0 1A EE B3 1E 40
12C0: D0 15 4C DF 10 A2 00 A0 C5
12C8: 00 C8 D0 FD E8 D0 F8 CE 64
12D0: BB 1E D0 03 4C 1B 10 4C DF
12D8: 2D 10 20 2F 14 20 2F 13 21
12E0: 4C F2 16 00 2D 40 2D 00 6F
12E8: D3 C0 D3 C0 D3 00 2D 40 E6
12F0: 2D 00 D3 20 FD 12 24 51 F9
12F8: 10 02 E6 50 60 85 4E 84 C1
1300: 4F A9 00 85 50 24 4E 10 50
1308: 05 38 A9 00 E5 4F 85 51 BC
1310: A2 08 06 51 26 50 06 4E 2C
1318: 90 11 18 A5 51 65 4F 85 6C
1320: 51 90 02 E6 50 24 4F 10 83
1328: 02 C6 50 CA D0 E4 60 A0 33
1330: 03 B1 FA D0 01 60 A0 04 7F
1338: B1 FA 8D A2 1E 0A 18 69 84
1340: 19 A8 B1 FA 85 FC C8 B1 66
1348: FA 85 FD A0 00 B1 FA 18 EB
1350: 71 FC 8D 9D 1E C8 B1 FA 6C
1358: 18 71 FC C9 07 90 05 EE 96
1360: 9D 1E E9 07 8D 9E 1E C8 76
1368: B1 FA 18 71 FC 8D 9F 1E BB
1370: AD 9E 1E 0A 0A 18 69 05 02
1378: A8 B1 FC 85 1C C8 B1 FC BB
1380: 85 1D C8 B1 FC 85 1E C8 E7
1388: B1 FC 85 1F A9 0A AC 98 D0
1390: 1E F0 03 18 69 07 A8 A9 46
1398: 01 91 FA C8 AD 9D 1E 91 41
13A0: FA C8 AD 9E 1E 91 FA C8 0C
13A8: AD 9F 1E 91 FA C8 AD A2 63
13B0: 1E 91 FA C8 B1 FA 85 EE A9
13B8: C8 B1 FA 85 EF A0 03 B1 21
13C0: FC 8D A0 1E C8 B1 FC 8D 53
13C8: A1 1E A9 00 8D B8 1E 8D 95
13D0: BA 1E AD 9F 1E 8D A3 1E 18
13D8: AD A1 1E 8D A4 1E 20 0F C7
13E0: 17 AC A0 1E 88 B1 FE 91 4E
13E8: EE 31 1E D1 1E F0 0A EE 6B
13F0: B8 1E C9 00 30 03 EE BA 5A
13F8: 1E 51 FE 11 1C 91 FE 88 21
1400: 10 E3 18 A5 EE 6D A0 1E 13
1408: 85 EE 90 02 E6 EF 18 A5 AD
1410: 1C 6D A0 1E 85 1C 90 02 57
1418: E6 1D 18 A5 1E 6D A0 1E 5E
1420: 85 1E 90 02 E6 1F EE A3 F9
1428: 1E CE A4 1E D0 B0 60 A9 3D
1430: 0A AC 98 1E F0 03 18 69 AA
1438: 07 A8 B1 FA D0 01 60 A9 E8
1440: 00 91 FA C8 B1 FA 8D 9D EA
1448: 1E C8 B1 FA 8D 9E 1E C8 83
1450: B1 FA 8D 9F 1E C8 B1 FA 2E
1458: 8D A2 1E C8 B1 FA 85 EE B3
1460: C8 B1 FA 85 EF AD A2 1E AA
1468: 0A 18 69 19 A8 B1 FA 85 E1
1470: FC C8 B1 FA 85 FD A0 03 97
1478: B1 FC 8D A0 1E C8 B1 FC E8
1480: 8D A1 1E AD 9F 1E 8D A3 AA
1488: 1E AD A1 1E 8D A4 1E 20 9C
1490: 0F 17 AC A0 1E 88 B1 EE 0B
1498: 91 FE 88 10 F9 18 A5 EE C5
14A0: 6D A0 1E 85 EE 90 02 E6 68
14A8: EF EE A3 1E CE A4 1E D0 F0
14B0: DE 60 A9 FF 85 1C A9 20 A5
14B8: 85 E6 8D 97 1E 20 F6 F3 DB
14C0: A9 00 8D A3 1E 8D 9D 1E 2A
14C8: 20 0F 17 A9 80 A0 27 91 A8
14D0: FE 88 10 FB EE A3 1E 20 BE
14D8: 0F 17 A9 80 A0 00 91 FE B2
14E0: A0 27 91 FE EE A3 1E AD 35
14E8: A3 1E C9 B8 D0 E9 20 0F AC
14F0: 17 A9 80 A0 27 91 FE 88 2F
14F8: 10 FB EE A3 1E AD A3 1E 4D
1500: C9 C0 D0 EA A9 00 8D 9D 0E
1508: 1E A9 F8 8D 9F 1E AD B3 28
1510: 1E 0A AA BD A8 16 85 1C C1
1518: BD A9 16 85 1D A0 00 8C 9E
1520: B7 1E AC B7 1E B1 1C F0 9F
1528: 66 EE B7 1E 48 8D B4 1E 1A
1530: 29 1F 8D B5 1E 68 29 60 09
1538: 8D B6 1E AD B6 1E F0 12 97
1540: C9 40 F0 1A 90 06 EE 9D 37
1548: 1E 4C 67 15 CE 9D 1E 4C 48
1550: 67 15 38 AD 9F 1E E9 08 A6
1558: 8D 9F 1E 4C 67 15 18 AD 27
1560: 9F 1E 69 08 8D 9F 1E 2C E2
1568: B4 1E 10 1B AD 9F 1E 8D DD
1570: A3 1E A9 80 A2 00 20 0F 95
1578: 17 A9 80 81 FE EE A3 1E D9
1580: A9 07 2C A3 1E D0 EF CE E3
1588: B5 1E D0 AF 4C 22 15 AC EB
1590: B7 1E C8 B1 1C 8D 9D 1E C2
1598: C8 B1 1C 8D A3 1E A9 B6 8F
15A0: 85 1C A9 16 85 1D A9 14 33
15A8: 8D A4 1E 20 0F 17 A0 02 A0
15B0: B1 1C 91 FE 88 10 F9 EE 44
15B8: A3 1E 18 A5 1C 69 03 85 AB
15C0: 1C 90 02 E6 1D CE A4 1E 57
15C8: D0 E1 A9 00 85 1C 85 FE AF
15D0: A9 20 85 1D A9 40 85 FF B3
15D8: A0 00 B1 1C 91 FE C8 D0 36
15E0: F9 E6 1D E6 FF A5 FF C9 34
15E8: 60 D0 EF 60 4B FF 86 CD B0
15F0: 00 07 7E 6A D1 A3 E6 66 A4
15F8: C6 90 A3 E6 65 86 C5 E3 42
1600: CC A3 E6 00 1D 0E 45 EC F0
1608: 47 25 EE 8B 6A 43 CD AA 5E
1610: C2 02 B4 45 65 00 05 96 D4
1618: 68 C3 45 A7 EB C3 83 E4 E6
1620: 83 C3 E6 C3 83 E2 A1 04 07
1628: 83 47 E6 83 C3 E3 C3 45 77
1630: E6 AC 83 C6 83 AA 83 C6 6C
1638: 83 AA 83 C6 45 65 83 00 74
1640: 1D 88 66 C6 A1 E3 A2 C8 01
1648: E1 27 E1 A1 66 C4 45 24 7A
1650: E1 63 05 EA C2 82 EA 81 0D
1658: C1 E5 46 81 05 E3 06 E4 68
1660: A1 C2 82 AD 81 C1 A6 C2 5C
1668: 82 A4 84 E1 A1 82 A1 E1 E9
1670: 81 68 05 C3 A1 E1 C3 EA 5B
```

```
1678: C1 81 E2 06 C1 00 19 0C EE
1680: 65 D3 E9 64 E2 81 C1 EE 67
1688: 8E A2 24 AA 24 A4 C9 E8 04
1690: 64 EC 00 0D 3F 46 F2 82 76
1698: E1 C2 EC C5 E2 C1 A2 C4 88
16A0: AC C2 A1 82 B1 00 05 94 5C
16A8: EC 15 95 16 F3 15 80 16 AF
16B0: 06 16 42 16 18 16 D5 AA 7E
16B8: D5 D5 AA D5 D5 AA 00 D5 27
16C0: AA 00 D5 AA 00 D5 AA 00 54
16C8: D5 AA 00 D5 AA 00 D5 AA 93
16D0: 00 D5 AA 00 D5 AA 00 D5 F6
16D8: AA 00 D5 AA 00 D5 AA 00 6C
16E0: D5 AA 00 D5 AA 00 D5 AA AB
16E8: 00 D5 AA 00 D5 AA D5 D5 BA
16F0: AA D5 AD 97 1E C9 40 A9 59
16F8: 00 2A AA BD 54 C0 8A 49 E4
1700: 01 8D 98 1E AD 97 1E 85 94
1708: E6 49 60 8D 97 1E 60 AD 84
1710: A3 1E 29 3F A8 B9 38 17 64
1718: 0D 97 1E 85 FF AD A3 1E EA
1720: 29 08 F0 02 A9 80 18 2C CE
1728: A3 1E 70 04 10 04 69 28 89
1730: 69 28 6D 9D 1E 85 FE 60 0A
1738: 00 04 08 0C 10 14 18 1C 46
1740: 00 04 08 0C 10 14 18 1C 4E
1748: 01 05 09 0D 11 15 19 1D 56
1750: 01 05 09 0D 11 15 19 1D 5E
1758: 02 06 0A 0E 12 16 1A 1E 66
1760: 02 06 0A 0E 12 16 1A 1E 6E
1768: 03 07 0B 0F 13 17 1B 1F 76
1770: 03 07 0B 0F 13 17 1B 1F 7E
1778: A9 00 A0 0A 91 FA A0 11 FA
1780: 91 FA A0 0F A5 EC 91 FA 3A
1788: C8 A5 ED 91 FA A0 09 B1 79
1790: FA 18 65 EC 85 EC 90 02 C0
1798: E6 ED A0 16 A5 EC 91 FA 2A
17A0: C8 A5 ED 91 FA A0 09 B1 91
17A8: FA 18 65 EC 85 EC 90 02 D8
17B0: E6 ED 60 A9 00 8D A2 1E 0E
17B8: AD A2 1E 0A 69 19 A8 B1 7D
17C0: FA 85 FC C8 B1 FA 85 FD 7C
17C8: 20 D8 17 EE A2 1E AD A2 9A
17D0: 1E A0 18 D1 FA D0 E1 60 95
17D8: A9 01 8D 9E 1E A0 03 B1 E2
17E0: FC 8D A0 1E C8 B1 FC 8D 7B
17E8: A1 1E A9 09 8D 99 1E A0 E4
17F0: 05 B1 FC 85 1C C8 B1 FC 6A
17F8: 85 1D AC 99 1E A5 EC 91 53
1800: FC C8 A5 ED 91 FC C8 8C 13
1808: 99 1E 20 36 18 A0 07 B1 F6
1810: FC 85 1C C8 B1 FC 85 1D D9
1818: AC 99 1E A5 EC 91 FC C8 93
1820: A5 ED 91 FC C8 8C 99 1E 6A
1828: 20 36 18 EE 9E 1E AD 9E 4F
1830: 1E C9 07 D0 BA 60 AD A1 24
1838: 1E 8D A4 1E A9 00 8D 9A 54
1840: 1E A0 00 A9 00 8D 9B 1E CD
1848: B1 1C 8D 9C 1E 0A AE 9E E8
1850: 1E 0A 2E 9B 1E CA D0 F9 49
1858: 2C 9C 1E 10 02 38 24 18 DB
1860: 6A 0D 9A 1E 91 EC AD 9B 75
1868: 1E 8D 9A 1E C8 CC A0 1E 19
1870: D0 D1 18 A5 1C 6D A0 1E D0
1878: 85 1C 90 02 E6 1D 18 A5 26
1880: EC 6D A0 1E 85 EC 90 02 7B
1888: E6 ED CE A4 1E D0 AD 60 BB
1890: A2 00 20 1E FB 98 60 2C 27
1898: 54 C0 2C 51 C0 20 58 FC F1
18A0: 2C C4 1E 30 3E A9 04 85 05
18A8: 25 20 22 FC A9 0F 85 24 40
18B0: A2 00 BD 0E 1A F0 06 20 8B
18B8: ED FD E8 D0 F5 AD BF 1E 8D
18C0: C9 0A .90 07 EE 17 06 E9 A4
18C8: 0A B0 F5 69 B0 8D 18 06 71
18D0: AD C0 1E C9 0A 90 07 EE F7
18D8: 15 06 E9 0A B0 F5 69 B0 D3
18E0: 8D 16 06 A9 00 8D C4 1E 96
18E8: 20 2A 19 A9 00 8D B3 1E 2D
18F0: AD 61 C0 30 05 8D A5 1E 33
18F8: 10 1D 2C A5 1E 30 18 8D C7
1900: A5 1E EE B3 1E AD B3 1E D2
1908: C9 07 D0 05 A9 00 8D B3 67
1910: 1E 18 69 B1 8D BF 04 A0 B3
1918: 00 A2 00 CA D0 FD 88 D0 FF
1920: FA AD 00 C0 10 CA 2C 10 5B
1928: C0 60 A2 00 BD 4E 19 F0 71
1930: 1C 85 25 20 22 FC E8 BD 0D
1938: 4E 19 85 24 E8 BD 4E 19 BE
1940: F0 07 E8 20 ED FD 4C 3D 09
1948: 19 E8 4C 2C 19 60 07 10 F5
1950: CD C9 C1 CD C9 A0 C9 C3 19
1958: C5 00 09 11 CC C5 D6 C5 90
1960: CC A0 B1 00 0C 09 D0 D2 50
1968: C5 D3 D3 A0 C1 CE D9 A0 94
1970: CB C5 D9 A0 D4 CF A0 C2 29
1978: C5 C7 C9 CE 00 00 2C 54 52
1980: C0 2C 51 C0 20 58 FC A2 53
1988: 00 BD FD 19 F0 1A 85 25 9B
1990: 20 22 FC E8 BD FD 19 85 27
1998: 24 E8 BD FD 19 F0 06 20 67
19A0: ED FD E8 D0 F5 E8 D0 E1 4A
19A8: AD B3 1E 18 69 B1 8D 96 A7
19B0: 07 18 AD BB 1E 69 B0 8D 63
19B8: BE 04 AD BF 1E C9 0A 90 B9
19C0: 07 EE BF 05 E9 0A B0 F5 49
19C8: 69 B0 8D C0 05 AD C0 1E 18
19D0: C9 0A 90 07 EE BD 05 E9 4F
19D8: 0A B0 F5 69 B0 8D BE 05 CF
19E0: A9 02 8D BC 1E A9 00 8D 0B
19E8: BD 1E 8D BE 1E CE BE 1E E6
19F0: D0 FB CE BD 1E D0 F6 CE 31
19F8: BC 1E D0 F1 60 07 10 CC 56
1A00: C5 D6 C5 CC 00 09 11 C3 5C
1A08: C1 D2 D3 00 0B 0F D3 C3 4C
1A10: CF D2 C5 A0 B0 B0 B0 B0 FD
1A18: 00 00 20 20 20 53 20 20 01
1A20: 32 45 20 40 20 20 20 20 A8
1A28: 20 00 20 20 20 20 20 7F B3
1A30: 20 20 08 43 1A 64 1A 85 CD
1A38: 1A A6 1A C7 1A E8 1A 09 94
1A40: 1B 2A 1B 01 03 00 03 11 2F
1A48: 4B 1B 7E 1B 20 35 20 20 A0
1A50: 20 20 20 20 20 20 20 46 AA
1A58: 2E 44 20 20 20 20 41 20 DE
1A60: 20 20 20 20 01 01 01 04 C4
1A68: 10 97 1C D7 1C 20 20 20 4D
1A70: 20 20 20 20 20 20 20 20 A4
1A78: 20 20 3A 20 20 20 20 20 EF
1A80: 20 4F 20 0F 23 01 04 08 BA
1A88: 04 08 17 1C 37 1C 20 30 FF
1A90: 00 20 30 20 20 36 20 20 0F
1A98: 20 20 49 20 20 45 20 20 86
1AA0: 20 20 20 20 20 20 01 01 77
1AA8: 06 04 10 17 1D 57 1D 34 09
1AB0: 20 31 20 20 20 2E 20 53 94
1AB8: 45 20 20 20 20 20 20 20 7F
1AC0: 20 20 20 20 20 20 00 01 95
1AC8: 03 06 03 11 B1 1B E4 1B 50
1AD0: 20 20 20 20 4F 20 55 20 E8
1AD8: 20 20 20 20 20 20 31 2E 3D
1AE0: 30 31 20 20 00 46 00 20 B8
1AE8: 00 06 06 04 10 17 1E 57 10
1AF0: 1E 00 20 20 20 20 3B 20 52
1AF8: 20 20 20 2E 7F 20 20 20 09
1B00: 24 20 20 20 7F 20 20 20 33
1B08: 20 00 06 08 04 08 57 1C 9A
1B10: 77 1C 20 20 20 41 20 44 99
1B18: 20 20 20 20 20 20 20 52 80
1B20: 20 20 7F 20 20 4F 20 20 FE
1B28: 20 20 00 06 01 04 10 97 A6
1B30: 1D D7 1D 7F 20 22 20 20 70
1B38: 20 20 20 20 20 20 20 20 6E
1B40: 20 20 20 20 46 20 20 20 A7
1B48: 49 20 20 D2 84 80 D5 8A B8
1B50: 80 D5 8A 80 D5 8A 80 D5 45
1B58: 8A 80 D5 8A 80 D5 8A 80 48
1B60: FD 8B 80 9F 8F 80 81 88 8C
1B68: 80 81 88 80 81 88 80 85 0D
1B70: 8A 80 D5 8A 80 D5 8A 80 60
1B78: D5 8A 80 89 89 80 FE 87 B8
1B80: 80 FF 8F 80 FF 8F 80 FF 30
1B88: 8F 80 FF 8F 80 FF 8F 80 43
1B90: FF 8F 80 FF 8F 80 FF 8F C8
1B98: 80 FF 8F 80 FF 8F 80 FF 48
1BA0: 8F 80 FF 8F 80 FF 8F 80 5B
1BA8: FF 8F 80 FF 8F 80 FF 8F E0
1BB0: 80 89 89 80 D5 8A 80 D5 72
1BB8: 8A 80 D5 8A 80 85 8A 80 67
1BC0: 81 88 80 81 88 80 81 88 D3
1BC8: 80 9F 8F 80 FD 8B 80 D5 16
1BD0: 8A 80 D5 8A 80 D5 8A 80 C0
1BD8: D5 8A 80 D5 8A 80 D5 8A 96
1BE0: 80 D3 84 80 FF 8F 80 FF 24
1BE8: 8F 80 FF 8F 80 FF 8F 80 A3
1BF0: FF 8F 80 FF 8F 80 FF 8F 29
1BF8: 80 FF 8F 80 FF 8F 80 FF A8
1C00: 8F 80 FF 8F 80 FF 8F 80 BC
1C08: FF 8F 80 FF 8F 80 FF 8F 42
1C10: 80 FF 8F 80 FE 87 80 A9 43
1C18: DD EA 80 AD F1 8A 81 AD 4F
1C20: E0 AA 81 A9 E0 AA 81 A9 9C
1C28: E0 AA 81 AD E0 AA 81 AD E8
1C30: F1 8A 81 A9 DD EA 80 FF 6A
1C38: FF FF 80 FF FF FF 81 FF 83
1C40: FF FF 81 FF FF FF 81 FF AB
1C48: FF FF 81 FF FF FF 81 FF B3
1C50: FF FF 81 FF FF FF 80 D6 90
1C58: BA 95 81 D1 8E B5 81 D5 C4
1C60: 86 B4 81 D5 86 94 81 D5 F5
1C68: 86 94 81 D5 86 B4 81 D1 72
1C70: 8E B5 81 D6 BA 95 81 FE 29
1C78: FF FF 81 FF FF FF 81 FF E3
1C80: FF FF 81 FF FF FF 81 FF EB
1C88: FF FF 81 FF FF FF 81 FF F3
1C90: FF FF 81 FE FF FF 81 00 EB
1C98: A0 85 80 00 A8 94 80 00 2B
1CA0: A8 95 80 00 AA D5 80 00 50
1CA8: AA C5 81 C0 BE D5 80 C0 F2
1CB0: B0 95 80 D0 E0 95 80 90 B2
1CB8: C0 85 80 94 C0 85 80 D2 FB
1CC0: 80 81 00 CA A0 81 00 D4 26
1CC8: AA 80 00 C8 AA 80 00 90 EA
1CD0: 8A 80 00 A0 8D 80 00 00 E6
1CD8: E0 87 80 00 F8 9F 80 00 BA
1CE0: F8 9F 80 00 FE FF 80 00 86
1CE8: FE FF 81 C0 FF FF 80 C0 9E
1CF0: FF 9F 80 F0 FF 9F 80 F0 A0
1CF8: FF 87 80 FC FF 87 80 FE 11
1D00: FF 81 00 FE FF 81 00 FC 8D
1D08: BF 80 00 F8 BF 80 00 F0 C2
1D10: 8F 80 00 E0 8F 80 00 A0 5F
1D18: 8D 80 00 90 8A 80 00 C8 61
1D20: AA 80 00 D4 AA 80 00 CA 3F
1D28: A0 81 00 D2 80 81 00 94 DE
1D30: C0 85 80 90 C0 85 80 D0 33
1D38: E0 95 80 C0 B0 95 80 C0 02
1D40: BE D5 80 00 AA C5 81 00 CE
1D48: AA D5 80 00 A8 95 80 00 F9
1D50: A8 94 80 00 A0 85 80 E0 11
1D58: 8F 80 00 F0 8F 80 00 F8 01
1D60: BF 80 00 FC BF 80 00 FE 69
1D68: FF 81 00 FE FF 81 00 FC F5
1D70: FF 87 80 F0 FF 87 80 F0 BB
1D78: FF 9F 80 C0 FF 9F 80 C0 F6
1D80: FF FF 80 00 FE FF 81 00 C5
1D88: FE FF 80 00 F8 9F 80 00 99
1D90: F8 9F 80 00 E0 87 80 D0 36
1D98: 82 80 00 94 8A 80 00 D4 A8
1DA0: 8A 80 00 D5 AA 80 00 D1 C6
1DA8: AA 80 00 D5 BE 81 00 D4 86
1DB0: 86 81 00 D4 83 85 80 D0 DF
1DB8: 81 84 80 D0 81 94 80 C0 12
1DC0: 80 A5 80 C0 82 A9 80 00 7C
1DC8: A4 95 80 00 AA 89 80 00 4A
1DD0: A8 84 80 00 D0 82 80 F0 13
1DD8: 83 80 00 FC 8F 80 00 FC 40
1DE0: 8F 80 00 FF BF 80 00 FF 03
1DE8: BF 80 00 FF FF 81 00 FC 26
1DF0: FF 81 00 FC FF 87 80 F0 6B
1DF8: FF 87 80 F0 FF 9F 80 C0 74
1E00: FF BF 80 C0 FF BF 80 00 48
1E08: FE 9F 80 00 FE 8F 80 00 F2
1E10: F8 87 80 00 F0 83 80 00 51
1E18: D0 82 80 00 A8 84 80 00 C5
1E20: AA 89 80 00 AA 95 80 C0 91
1E28: 82 A9 80 C0 80 A5 80 D0 98
1E30: 81 94 80 D0 81 84 80 D4 63
1E38: 83 85 80 D4 86 81 00 D5 05
1E40: BE 81 00 D1 AA 80 00 D5 86
1E48: AA 80 00 D4 8A 80 00 94 32
1E50: 8A 80 00 D0 82 80 00 00 15
1E58: F0 83 80 00 F8 87 80 00 E4
1E60: FE 8F 80 00 FE 9F 80 C0 48
1E68: FF BF 80 C0 FF BF 80 F0 A1
1E70: FF 9F 80 F0 FF 87 80 FC CF
1E78: FF 87 80 FC FF 81 00 FF 7C
1E80: FF 81 00 FF BF 80 00 FF 1D
1E88: BF 80 00 FC 8F 80 00 FC 10
1E90: 8F 80 00 F0 83 80 00 00 E1
```

©

proceeding to the more populous areas lying southward. Are you up to the challenge?

Though it's not a particularly long program, "UFO Invasion" offers quite a test for your gaming skills, particularly at the higher levels. Type in the version for your computer and save a copy before you run it. As the screen photos illustrate, both versions look and play almost identically.

Since the Amiga version doesn't use line numbers, we've used a special character (a left arrow) to show you where each program line ends. Don't try to type in the arrows—they're present only to show you where each line ends. (Actually, you can't type an arrow even if you want to, since we deliberately picked a symbol that's not available on the Amiga keyboard.) Instead, wherever you see a left arrow in the listing, you should press RETURN or move the cursor off the line to enter it into memory. To illustrate, look at these program lines:

```
DEFINT A-Z←
RANDOMIZE TIMER←
SCREEN 1, 320, 200, 2, 1←
WINDOW 1,"UFO Invasion",(0,0)-(3
11,185),20,1←
```

The first three lines are short enough to fit into one magazine column, but the fourth program line is so long that it wraps around onto a second line. The arrow shows you that the line ends after the final 1, not after the 3.

First Line Of Defense

When UFO Invasion begins, you'll hear the sound of an alarm siren and see two warning messages scroll across the screen. The middle of the screen contains your control panel. The observatory window at the top gives you a direct view of the skyline in your defense sector. Within the window is the aiming crosshair for your missile launcher, and directly below is a radar screen.

When the saucer-shaped UFO appears, your job is to move the crosshair onto the UFO (using the cursor keys) and launch a missile at it (by pressing the space bar). If your missile hits the UFO, the automated craft is vaporized immediately.

Before it can fire at your base, the UFO must locate your position. Once your position is located, the UFO is certain to hit the mark. Your force shields are powerful enough to protect you against three hits by the UFO, but the fourth hit neutralizes your defenses and paves the way for a successful invasion (ending the game as well).

Control Panel

The control panel is equipped with six gauges to help you monitor events. On each side of the circular radar screen are two ladder gauges. The gauge at the far right shows you how many UFOs remain to be eliminated in the current level. There are eight levels in all; you must eliminate 29 UFOs at each level before advancing to the next.

The gauge directly to the right of the radar screen indicates how close the UFO is to locating your position. When this indicator reaches the top, the UFO scores a hit.

The gauge at the far left shows your points for the current level. You receive 100 points (shown as one bar on the ladder) for each UFO you destroy, with an additional bar for hitting the UFO before the timer is halfway to the top. If you score two bars for every UFO on the current level, you receive a bonus equal to 1,000 points times the level number.

Directly underneath the radar screen are two additional indicators that show you how many levels have been completed, and how many hits your shields have sustained.

Press the cursor keys to move the aiming crosshair left, right, up, or down. To fire a missile, press the space bar. You can quit the game at any time by pressing Q. In levels 1, 2, 4, 6, and 8 you can view the UFOs through the observatory window. In levels 3, 5, and 7 the sky is obscured by a thick cloud cover, forcing you to guide the missiles by radar alone. The radar screen shows the position of the UFO in relation to your aiming crosshair. Aim with the cursor keys until the red dot is in the center of the radar panel, then fire.

Amiga Version

Converting the original PC/PCjr game to Amiga BASIC was a very easy project. First, we used modems to transfer the PC program text to the Amiga over the phone line. Then we changed a few lines that were obviously unusable in Amiga BASIC (those with KEY and PLAY statements). In less than an hour, after changing about a dozen program lines, we had the PC game running on the Amiga—a testament to the close similarity between the BASICs on both machines.

Though the two programs look very different on the surface, the differences are largely cosmetic. To improve the Amiga program's readability and make it easier to type, we stripped off the line numbers, substituted meaningful labels where needed, and chopped most multi-statement lines into single-statement lines. Line numbers are unnecessary in Amiga BASIC; statements like *GOSUB PrintMessage* and *GOTO MainLoop* are much easier to understand than number-oriented statements like *GOSUB 890*. And in most cases there's little to gain by "crunching" multiple statements onto one line. We made no efforts to speed up the Amiga version, yet because of the Amiga's speedier processor, this program runs much faster than the PC/PCjr game.

If you compare the two programs statement by statement, you'll see that they're still nearly identical. Of course, the Amiga needs SCREEN and WINDOW statements to create a graphics screen equivalent to the original PC/PCjr screen. Since ON KEY and PLAY don't exist in Amiga BASIC, substitutes had to be found there as well (we used INKEY$ to read the keyboard and SOUND for sound effects). But the meat of the program—high-resolution drawing with LINE statements and animation with GET and PUT—is exactly the same.

If you're a PC owner who just bought an Amiga, or an Amiga owner looking for more type-in programs, this project shows how simple it can be to convert programs from IBM BASIC to Amiga BASIC. (Another language which is even more similar to Amiga BASIC is Microsoft BASIC for the Macintosh.) As a general rule, any game that relies chiefly on LINE, GET, and PUT should transfer from the

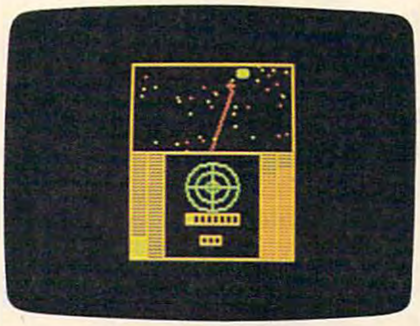
PC to the Amiga quite easily. Just be sure to set up the right sort of screen at the beginning of the program.

To highlight the similarity between the two versions of BASIC, we did not add many machine-specific features to the Amiga version. However, you may find it interesting to add some extra features of your own. For instance, why not add voice synthesis to the messages that scroll across the screen? If you have a stereo hookup, you might want to modify the sound routines to take advantage of the Amiga's stereo sound capabilities. The Amiga version of "Switchbox" (COMPUTE!, March 1986) contains examples of how to do both, as well as other tips on writing games in Amiga BASIC. On a larger scale, you might want to try enlarging the playfield. In the original PC/PCjr version, the game screen is kept quite small to make the game run faster. But Amiga BASIC is fast enough to permit convincing animation within a much larger area.

Program 1: UFO Invasion For IBM PC/PCjr

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
LA 1Ø REM ---------- INITIALIZE
      VARIABLES ----------
PN 2Ø SCREEN 1:COLOR Ø,Ø,Ø:CLS:K
      EY OFF:RANDOMIZE TIMER:PLA
      Y "mb"
BF 3Ø DIM SH(2ØØØ),UFO(2ØØ),GD(2
      ØØ),X(3Ø),Y(3Ø),RADAR(5Ø)
IG 4Ø L=1:TL=8:LIVES=Ø:SCORE=Ø:R
      X=1ØØ/15:RY=5Ø/12
EL 5Ø REM ---------- SET UP SCRE
      EN ----------
IJ 6Ø LINE (111,51)-(211,159),3,
      B:LINE (111,1Ø1)-(211,1Ø1)
      ,3
NC 7Ø LINE (121,1Ø1)-(121,159),3
      :LINE (131,1Ø1)-(131,159),
      3
AF 8Ø LINE (191,1Ø1)-(191,159),3
      :LINE (2Ø1,1Ø1)-(2Ø1,159),
      3
JP 9Ø FOR Y=157 TO 1Ø3 STEP -2
FF 1ØØ LINE (111,Y)-(131,Y),3:LI
       NE (191,Y)-(211,Y),3
HJ 11Ø NEXT Y
HL 12Ø CIRCLE (161,12Ø),16,1:CIR
       CLE (161,12Ø),1Ø,1:CIRCLE
        (161,12Ø),4,1
MO 13Ø LINE (146,12Ø)-(176,12Ø),
       1:LINE (161,1Ø8)-(161,132
       ),1
IB 14Ø FOR X=145 TO 173 STEP 4:L
       INE (X,135)-(X+4,14Ø),3,B
       :NEXT X
DA 15Ø FOR X=154 TO 162 STEP 4:L
       INE (X,147)-(X+4,151),3,B
       :NEXT X
EJ 16Ø LINE (1,Ø)-(6,Ø),1:LINE (
       Ø,1)-(7,1),1:LINE (1,2)-(
       7,2),3
IJ 17Ø LINE (Ø,3)-(7,3),1:LINE (
       1,4)-(6,4),1:GET (Ø,Ø)-(7
       ,6),UFO
JM 18Ø LINE (Ø,Ø)-(7,7),Ø,BF
LE 19Ø LINE (Ø,3)-(4,3),2:LINE (
       2,1)-(2,5),2:GET (Ø,Ø)-(5
       ,5),GD
IN 2ØØ LINE (Ø,Ø)-(7,7),Ø,BF
EM 21Ø GOSUB 27Ø:B$=" ENEMY ALER
       T":GOSUB 25Ø
BE 22Ø B$=" UFO INVASION":GOSUB
       25Ø
LA 23Ø LINE (Ø,Ø)-(1,1),2,BF:GET
        (Ø,Ø)-(1,1),RADAR
```



The IBM PC/PCjr version of "UFO Invasion" pits you, the lonely defender in an Arctic wasteland, against waves of oncoming robot craft.

```
DP 24Ø GOTO 32Ø
PA 25Ø FOR I=1 TO 39:LOCATE 1,I:
       FOR SA=1 TO 2Ø:NEXT SA:PR
       INT LEFT$(B$,4Ø-I):NEXT I
NI 26Ø RETURN
HK 27Ø FOR I=1 TO 1Ø
KC 28Ø FOR P=1ØØØ TO 19ØØ STEP 2
       5:SOUND P,.2:NEXT P
OK 29Ø NEXT I
MN 3ØØ RETURN
JA 31Ø REM ---------- SET UP KEY
       BOARD ----------
KD 32Ø DEF SEG=Ø:POKE 1Ø47,PEEK(
       1Ø47) OR 64
DC 33Ø ON KEY (11) GOSUB 63Ø:KEY
        (11) ON
IL 34Ø ON KEY (12) GOSUB 64Ø:KEY
        (12) ON
ME 35Ø ON KEY (13) GOSUB 65Ø:KEY
        (13) ON
BN 36Ø ON KEY (14) GOSUB 66Ø:KEY
        (14) ON
JN 37Ø KEY 15,CHR$(&H4Ø)+CHR$(&H
       39):ON KEY (15) GOSUB 67Ø
       :KEY (15) ON
OE 38Ø KEY 16,CHR$(&H4Ø)+CHR$(&H
       1Ø):ON KEY (16) GOSUB 68Ø
       :KEY (16) ON
CB 39Ø REM ---------- START A NE
       W LEVEL ----------
CC 4ØØ B=16Ø:PTS=Ø:T=Ø
IG 41Ø FOR S=158 TO 1Ø2 STEP -2:
       LINE (2Ø2,S)-(21Ø,S),2:NE
       XT S
IC 42Ø IF L=3 OR L=5 OR L=7 THEN
        CLR=Ø:GOTO 44Ø ELSE CLR=
       1
MA 43Ø FOR S=1 TO 6Ø:PSET(112+RN
       D*98,52+RND*48),INT(RND*4
       ):NEXT S
EA 44Ø S=1ØØ
FF 45Ø XU=112+RND*9Ø:YU=52+RND*4
       Ø:IF CLR THEN PUT(XU,YU),
       UFO,XOR
FF 46Ø XG=16Ø:YG=75:PUT(XG,YG),G
       D,XOR
QJ 47Ø LINE(142+L*4,136)-(144+L*
       4,139),1,BF
OC 48Ø REM ---------- PERFORM MA
       IN LOOP ----------
CD 49Ø GOSUB 7ØØ ' MOVE CROSS HA
       IRS
LO 5ØØ GOSUB 55Ø ' MOVE UFO
BD 51Ø IF FIRED THEN GOSUB 77Ø:I
       F S=158 THEN 1ØØØ
ON 52Ø T=T+1:IF T>TL THEN B=B-2:
       T=Ø:LINE(192,B)-(2ØØ,B),2
       :IF B=1Ø2 THEN GOSUB 119Ø
        ' CHECK TIME
IH 53Ø GOTO 49Ø
KK 54Ø REM ---------- MOVE UFO &
       RADAR ----------
KF 55Ø IF RND<.1 THEN CXU=RND*1Ø
       -5:CYU=RND*6-3
PD 56Ø IF CLR THEN PUT (XU,YU),U
       FO,XOR
LM 57Ø XU=XU+CXU:IF XU>2ØØ THEN
       XU=2ØØ ELSE IF XU<112 THE
       N XU=112
GN 58Ø YU=YU+CYU:IF YU>9Ø THEN Y
       U=9Ø ELSE IF YU<52 THEN Y
       U=52
QJ 59Ø IF CLR THEN PUT (XU,YU),U
       FO,XOR
LF 6ØØ PUT(XR,YR),RADAR,XOR:XR=1
       61+(XU-XG)/RX:YR=12Ø+(YU-
       YG)/RY:PUT (XR,YR),RADAR,
       XOR
MC 61Ø RETURN
FE 62Ø REM ---------- RESPOND TO
       KEY PRESSES ----------
KP 63Ø CYG=CYG-5:RETURN
JB 64Ø CXG=CXG-5:RETURN
IL 65Ø CXG=CXG+5:RETURN
JN 66Ø CYG=CYG+5:RETURN
OB 67Ø FIRED=1:RETURN
LD 68Ø B$=" GAME STOPPED":SCORE=
       SCORE+PTS*1ØØ*L:RETURN 13
       9Ø
KM 69Ø REM ---------- MOVE CROSS
       HAIRS ----------
DO 7ØØ PUT (XG,YG),GD,XOR
LO 71Ø XG=XG+CXG:IF XG>2ØØ THEN
       XG=2ØØ ELSE IF XG<112 THE
       N XG=112
LD 72Ø YG=YG+CYG:IF YG>9Ø THEN Y
       G=9Ø ELSE IF YG<52 THEN Y
       G=52
KH 73Ø CXG=Ø:CYG=Ø
DG 74Ø PUT (XG,YG),GD,XOR
NL 75Ø RETURN
LA 76Ø REM ---------- FIRE -----
       -----
NP 77Ø PLAY "L64 T255 BAGFEDC <B
       AGFEDC>"
QJ 78Ø IF CLR THEN PUT (XU,YU),U
       FO,XOR
MG 79Ø LINE (16Ø,1ØØ)-(XG+3,YG+6
       ),2
GB 8ØØ FIRED=Ø : LINE (16Ø,1ØØ)-
       (XG+3,YG+6),Ø
IC 81Ø IF XG+3>XU AND XG+3<XU+9
       AND YG+4>YU AND YG+3<YU+6
        THEN 85Ø
PO 82Ø IF CLR THEN PUT (XU,YU),U
       FO,XOR
NI 83Ø RETURN
JE 84Ø REM ---------- UFO IS HIT
       ----------
EJ 85Ø PUT (XG,YG),GD,XOR
CP 86Ø FOR E=1 TO 3Ø:X(E)=XU+RND
       *6+1:Y(E)=YU+RND*6+1:PSET
        (X(E),Y(E)),2:SOUND 8Ø,.
       1:NEXT E
NG 87Ø FOR E=1 TO 3Ø:PRESET (X(E
       ),Y(E)):NEXT E
EL 88Ø IF CLR THEN PSET (XU+RND*
       6+1,YU+RND*6+1),INT(RND*4
       )
GB 89Ø XU=112+RND*9Ø:YU=52+RND*4
       Ø:IF CLR THEN PUT(XU,YU),
```

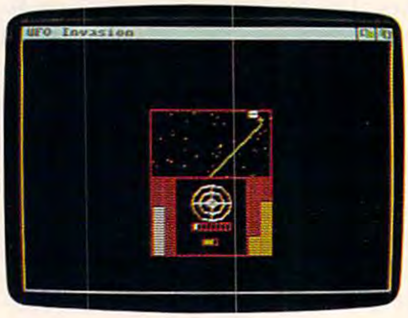
```
           UFO,XOR
KI 9ØØ  PUT(XG,YG),GD,XOR
FJ 91Ø  REM ----------- ADD SCORE
           -----------
DB 92Ø  S=S+2:LINE(2Ø2,S)-(21Ø,S)
           ,Ø
HN 93Ø  PTS=PTS+1:GOSUB 97Ø
DF 94Ø  IF B>13Ø THEN PTS=PTS+1:G
           OSUB 97Ø
EJ 95Ø  FOR X=B TO 16Ø STEP 2:LIN
           E(192,X)-(2ØØ,X),Ø:NEXT X
           :B=16Ø
NP 96Ø  RETURN
GE 97Ø  HX=112:HY=16Ø-PTS*2:IF PT
           S>29 THEN HX=122:HY=16Ø-(
           PTS-29)*2
QL 98Ø  LINE (HX,HY)-(HX+8,HY),1
NF 99Ø  RETURN
BH 1ØØØ REM ----------- ALL UFOS
           DESTROYED - LEVEL COMPLE
           TED -----------
IP 1Ø1Ø L=L+1:TL=TL-1
BO 1Ø2Ø FOR P=1Ø2 TO 158 STEP 2:
           LINE(112,P)-(12Ø,P),Ø:LI
           NE(122,P)-(13Ø,P),Ø:NEXT
           P
BI 1Ø3Ø LINE(112,52)-(2Ø9,99),Ø,
           BF
LI 1Ø4Ø SCORE=SCORE+PTS*(L-1)*1Ø
           Ø
ON 1Ø5Ø IF PTS<58 THEN 114Ø
DD 1Ø6Ø PLAY "O3T12Ø L8GGL16EEL8
           EL16D+EL8EP8 L16EEL8EL16
           EEL8GL16EGL4FL8DP8"
PO 1Ø7Ø PLAY "DDL16C+DL8DL16C+DL
           8FP4 L16ED+EL8GP16G16A8A
           8D3P4
KI 1Ø8Ø B$=" YOU PASSED LEVEL "+
           STR$(L-1):GOSUB 25Ø
EM 1Ø9Ø PLAY "O3T12Ø L8GGL16EEL8
           EL16D+EL8EP8 L16EEL8EL16
           EEL8GL16EGL4FL8DP8"
NA 11ØØ PLAY "DDL16C+DL8DL16C+DL
           4FP8L16CDEGP8L16CDEGP8 L
           16CDED4C4"
JC 111Ø B$=STR$(1ØØØ*(L-1))+" PO
           INTS BONUS":GOSUB 25Ø
KF 112Ø FOR I=1 TO 1ØØØ:NEXT I
FM 113Ø SCORE=SCORE+1ØØØ*(L-1)
OF 114Ø IF L>8 THEN 137Ø
IE 115Ø B$=" LEVEL "+STR$(L):GOS
           UB 25Ø
FD 116Ø FOR I=1 TO 5ØØ:NEXT I
QO 117Ø GOTO 4ØØ
JD 118Ø REM ----------- TIME'S UP
            - TAKE A HIT -----------
MA 119Ø XB=XU+4:YB=YU+6:IF CLR T
           HEN PUT (XU,YU),UFO,XOR
CJ 1195 LINE(XB,YB)-(112,1ØØ),2:
           LINE (XB,YB)-(21Ø,1ØØ),2
KJ 12ØØ LINE (XB,YU)-(112,52),2:
           LINE (XB,YU)-(21Ø,52),2
FN 121Ø LINE(XB,YB)-(112,1ØØ),Ø:
           LINE (XB,YB)-(21Ø,1ØØ),Ø
QP 122Ø LINE (XB,YU)-(112,52),Ø:
           LINE (XB,YU)-(21Ø,52),Ø
BM 1225 IF CLR THEN PUT (XU,YU),
           UFO,XOR
GL 123Ø FOR I=1 TO 2Ø
DB 124Ø COLOR Ø,1
IA 125Ø PLAY "L64 T255 BAGFEDC <
           BAGFEDC>"
DJ 126Ø COLOR Ø,Ø
HI 127Ø NEXT I
PK 128Ø PTS=PTS-2:IF PTS<Ø THEN
           PTS=Ø
BE 129Ø IF LIVES=3 THEN 134Ø
GD 13ØØ FOR X=1Ø2 TO 158 STEP 2:
           LINE(192,X)-(2ØØ,X),Ø:NE
           XT X:B=16Ø
GB 131Ø LIVES=LIVES+1:LINE (151+
           4*LIVES,148)-(153+4*LIVE
           S,15Ø),2,BF
IH 132Ø RETURN
EI 133Ø REM ----------- UFOS WIN
           -----------
MM 134Ø SOUND 13Ø,Ø:COLOR Ø,1
QO 135Ø B$=" GAME OVER":GOTO 139
           Ø
MJ 136Ø REM ----------- SAM WINS
           -----------
KB 137Ø B$=" YOU WIN !!!":GOSUB
           146Ø
LP 138Ø GOSUB 146Ø
KH 139Ø GOSUB 25Ø
BE 14ØØ C$=" SCORE"+STR$(SCORE)
DC 141Ø FOR I=1 TO 15:LOCATE 1,I
           :PRINT LEFT$(C$,4Ø-I):NE
           XT I
CH 142Ø IF INKEY$<>"" THEN 142Ø
IN 143Ø LOCATE 23,7:INPUT "ENTER
            Y TO PLAY AGAIN:  ";R$
DN 144Ø IF R$="Y" THEN RUN
II 145Ø END
BB 146Ø PLAY "T8ØO3ML C4G4F16E16
           D16O4C4 O3G4F16E16D16O4C
           4"
OP 147Ø PLAY "O3G4F16E16F16D4MSC
           4"
KL 148Ø RETURN
```

Program 2: UFO Invasion For Amiga

Version by Philip I. Nelson, Assistant Editor

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
◄
DEFINT A-Z◄
RANDOMIZE TIMER◄
SCREEN 1, 320, 2ØØ, 2, 1◄
WINDOW 1,"UFO Invasion",(Ø,Ø)-(3
11,185),2Ø,1◄
PALETTE Ø,Ø,Ø,Ø◄
PALETTE 1,1,1,1◄
PALETTE 2,Ø,1,Ø◄
PALETTE 3,1,Ø,Ø◄
CLS◄
DIM SH(2ØØØ),UFO(2ØØ),GD(2ØØ)◄
DIM X(3Ø),Y(3Ø),Radar(5Ø)◄
L=1: TL=8: Lives=Ø: Score=Ø◄
RX=1ØØ/15: RY=5Ø/12◄
LINE (111,51)-(211,159),3,B◄
LINE (111,1Ø1)-(211,1Ø1),3◄
LINE (121,1Ø1)-(121,159),3◄
LINE (131,1Ø1)-(131,159),3◄
LINE (191,1Ø1)-(191,159),3◄
LINE (2Ø1,1Ø1)-(2Ø1,159),3◄
FOR Y=157 TO 1Ø3 STEP -2◄
LINE (111,Y)-(131,Y),3◄
LINE (191,Y)-(211,Y),3◄
NEXT◄
CIRCLE (161,12Ø),16,1◄
CIRCLE (161,12Ø),1Ø,1◄
CIRCLE (161,12Ø),4,1◄
LINE (146,12Ø)-(176,12Ø),1◄
LINE (161,1Ø8)-(161,132),1◄
FOR X=145 TO 173 STEP 4◄
LINE (X,135)-(X+4,14Ø),3,B◄
NEXT◄
FOR X=154 TO 162 STEP 4◄
LINE (X,147)-(X+4,151),3,B◄
NEXT◄
LINE (1,Ø)-(6,Ø),1◄
LINE (Ø,1)-(7,1),1◄
LINE (1,2)-(7,2),3◄
LINE (Ø,3)-(7,3),1◄
LINE (1,4)-(6,4),1◄
GET (Ø,Ø)-(7,6),UFO◄
LINE (Ø,Ø)-(7,7),Ø,bf◄
LINE (Ø,3)-(4,3),2◄
LINE (2,1)-(2,5),2◄
GET (Ø,Ø)-(5,5),GD◄
LINE (Ø,Ø)-(7,7),Ø,bf◄
LINE (Ø,Ø)-(1,1),2,bf◄
GET (Ø,Ø)-(1,1),Radar◄
PUT (Ø,Ø),Radar◄
GOSUB Siren◄
B$=" Enemy Alert "◄
GOSUB PrintMessage◄
GOSUB Siren◄
B$=" UFO Invasion "◄
GOSUB PrintMessage◄
GOTO NewLevel◄
◄
◄
PrintMessage:◄
FOR J=1 TO 39◄
LOCATE 1,J◄
SOUND 4ØØ+(J*1Ø),.1◄
PRINT LEFT$(B$,4Ø-J)◄
NEXT◄
RETURN◄
◄
◄
Siren:◄
FOR J=1 TO 1Ø◄
FOR P=1ØØØ TO 19ØØ STEP 55◄
SOUND P,.2◄
NEXT◄
NEXT◄
RETURN◄
```



This photo illustrates how similar the Amiga version of "UFO Invasion" is to the PC/PCjr game. About 90 percent of the code is identical to the original program.

```
◄
◄
NewLevel:◄
B=16Ø: Pts=Ø: T=Ø◄
FOR S=158 TO 1Ø2 STEP -2◄
LINE (2Ø2,S)-(21Ø,S),2◄
NEXT◄
IF L=3 OR L=5 OR L=7 THEN clr=Ø:
GOTO Mip ELSE clr=1◄
FOR S=1 TO 6Ø◄
PSET(112+RND*98,52+RND*48),INT(R
ND*4)◄
SOUND 15ØØ+INT(RND(1)*1ØØØ),.1◄
NEXT◄
Mip:◄
S=1ØØ: Xu=112+RND*9Ø◄
Yu=52+RND*4Ø◄
IF clr THEN PUT(Xu,Yu),UFO◄
xg=16Ø: yg=75◄
PUT(xg,yg),GD◄
LINE(142+L*4,136)-(144+L*4,139),
1,bf◄
◄
◄
MainLoop:◄
X$=INKEY$: IF UCASE$(X$)="Q" THE
N Quit◄
IF X$="" OR X$<CHR$(28) OR X$>CH
R$(32) THEN Skip◄
ON ASC(X$)-27 GOTO Up, Down, Rig
ht, Left, Hit◄
GOTO Skip◄
Up:◄
Cyg=Cyg-5:GOTO Skip◄
Down:◄
Cyg=Cyg+5:GOTO Skip◄
Right:◄
Cxg=Cxg+5:GOTO Skip◄
Left:◄
Cxg=Cxg-5:GOTO Skip◄
Hit:◄
```

```
Fired=1◄
◄
◄
Skip:◄
PUT (xg,yg),GD◄
xg=xg+Cxg◄
IF xg>200 THEN xg=200 ELSE IF xg
<112 THEN xg=112◄
yg=yg+Cyg◄
IF yg>90 THEN yg=90 ELSE IF yg<5
2 THEN yg=52◄
Cxg=0:Cyg=0◄
PUT (xg,yg),GD◄
IF RND<.1 THEN Cxu=RND*10-5:Cyu=
RND*6-3◄
IF clr THEN PUT (Xu,Yu),UFO◄
Xu=Xu+Cxu◄
IF Xu>200 THEN Xu=200 ELSE IF Xu
<112 THEN Xu=112◄
Yu=Yu+Cyu◄
IF Yu>90 THEN Yu=90 ELSE IF Yu<5
2 THEN Yu=52◄
IF clr THEN PUT (Xu,Yu),UFO◄
IF NotFirst THEN PUT(Xr,Yr),Rada
r◄
NotFirst=1◄
Xr=161+(Xu-xg)/RX◄
Yr=120+(Yu-yg)/RY◄
PUT (Xr,Yr),Radar◄
IF Fired THEN GOSUB Shoot:IF S=1
58 THEN AllGone◄
T=T+1◄
IF T>TL THEN B=B-2:T=0:LINE(192,
B)-(200,B),2:IF B=102 THEN GOSUB
TakeShot◄
GOTO MainLoop◄
◄
◄
Quit:◄
B$=" Game stopped"◄
Score=Score+Pts*100*L◄
GOTO GameOver◄
◄
◄
Shoot:◄
IF clr THEN PUT (Xu,Yu),UFO◄
LINE (160,100)-(xg+3,yg+6),2◄
FOR J=50 TO 1000 STEP 200◄
SOUND J,.1◄
NEXT◄
Fired=0◄
LINE (160,100)-(xg+3,yg+6),0◄
IF xg+3>Xu AND xg+3<Xu+9 AND yg+
4>Yu AND yg+3<Yu+6 THEN HitUFO◄
IF clr THEN PUT (Xu,Yu),UFO◄
RETURN◄
◄
◄
HitUFO:◄
PUT (xg,yg),GD◄
FOR e=1 TO 30◄
X(e)=Xu+RND*6+1◄
Y(e)=Yu+RND*6+1◄
PSET (X(e),Y(e)),2◄
SOUND 820,.1◄
NEXT◄
FOR e=1 TO 30◄
PRESET (X(e),Y(e))◄
NEXT◄
IF clr THEN PSET (Xu+RND*6+1,Yu+
RND*6+1),INT(RND*4)◄
Xu=112+RND*90◄
Yu=52+RND*40◄
IF clr THEN PUT(Xu,Yu),UFO◄
PUT(xg,yg),GD◄
AddScore:◄
S=S+2◄
LINE(202,S)-(210,S),0◄
Pts=Pts+1◄
GOSUB Here◄
IF B>130 THEN Pts=Pts+1:GOSUB He
re◄
FOR X=B TO 160 STEP 2◄
LINE(192,X)-(200,X),0◄
NEXT◄
B=160◄
RETURN◄
◄
◄
Here:◄
Hx=112: Hy=160-Pts*2◄
IF Pts>29 THEN Hx=122:Hy=160-(Pt
s-29)*2◄
LINE (Hx,Hy)-(Hx+8,Hy),1◄
RETURN◄
◄
◄
AllGone:◄
L=L+1: TL=TL-1◄
FOR P=102 TO 158 STEP 2◄
LINE(112,P)-(120,P),0◄
LINE(122,P)-(130,P),0◄
NEXT◄
LINE(112,52)-(209,99),0,bf◄
Score=Score+Pts*(L-1)*100◄
IF Pts<58 THEN GOTO There◄
FOR J=500 TO 2500 STEP 500◄
SOUND J,2◄
NEXT◄
B$=" You passed Level "+STR$(L-1
)◄
GOSUB PrintMessage◄
B$=STR$(1000*(L-1))+" Points bon
us"◄
GOSUB PrintMessage◄
FOR J=1 TO 2000◄
NEXT◄
Score=Score+1000*(L-1)◄
IF L=9 THEN PlayerWins◄
There:◄
IF L>8 THEN PlayerWins◄
B$=" Level "+STR$(L)◄
GOSUB PrintMessage◄
FOR J=1 TO 1500◄
NEXT◄
GOTO NewLevel◄
◄
◄
TakeShot:◄
PUT (xg,yg),GD◄
IF clr THEN PUT (Xu,Yu),UFO◄
Xb=Xu+4: Yb=Yu+6◄
LINE(Xb,Yb)-(112,100),2◄
LINE (Xb,Yb)-(210,100),2◄
LINE (Xb,Yu)-(112,52),2◄
LINE (Xb,Yu)-(210,52),2◄
LINE(Xb,Yb)-(112,100),0◄
LINE (Xb,Yb)-(210,100),0◄
LINE (Xb,Yu)-(112,52),0◄
LINE (Xb,Yu)-(210,52),0◄
IF clr THEN PUT (Xu,Yu),UFO◄
PUT (xg,yg),GD◄
PALETTE 0,1,0,0◄
FOR K=400 TO 500◄
SOUND K,.1◄
NEXT◄
PALETTE 0,0,0,0◄
Pts=Pts-2◄
IF Pts<0 THEN Pts=0◄
IF Lives=3 THEN UFOgotcha◄
FOR X=102 TO 158 STEP 2◄
LINE(192,X)-(200,X),0◄
NEXT◄
B=160◄
Lives=Lives+1◄
LINE (151+4*Lives,148)-(153+4*Li
ves,150),2,bf◄
RETURN◄
◄
◄
UFOgotcha:◄
PALETTE 0,1,0,0◄
FOR J=400 TO 500 STEP 3◄
SOUND J,.1◄
NEXT◄
FOR J=500 TO 400 STEP -3◄
SOUND J,.1◄
NEXT◄
PALETTE 0,0,0,0◄
B$=" Game Over"◄
GOTO GameOver◄
◄
◄
PlayerWins:◄
B$=" You win!"◄
GOSUB WinSound◄
◄
◄
GameOver:◄
GOSUB PrintMessage◄
c$=" Score"+STR$(Score)◄
FOR J=1 TO 15◄
LOCATE 1,J◄
PRINT LEFT$(c$,40-J)◄
NEXT◄
CleanBuffer:◄
IF INKEY$<>"" THEN CleanBuffer◄
LOCATE 23,9◄
PRINT "Press Y to play again";◄
X$=""◄
WHILE X$=""◄
X$=INKEY$◄
WEND◄
IF UCASE$(X$)="Y" THEN RUN◄
CLS: END◄
◄
◄
WinSound:◄
FOR J=1 TO 2◄
RESTORE MusicData◄
SoundLoop:◄
READ X◄
IF X<>65535& THEN◄
SOUND X,1◄
GOTO SoundLoop◄
END IF◄
NEXT◄
SOUND 550, 8◄
RETURN◄
◄
◄
MusicData:◄
DATA 550, 500, 450, 400, 350◄
DATA 550, 500, 450, 400, 350◄
DATA 300, 350, 400, 450, 500◄
DATA 65535◄
◄
```

©

Skyfox For Commodore And Apple

Richard Mansfield, Senior Editor

Requirements: Commodore 64 with a joystick; Apple II-series computer with at least 64K RAM; Apple Macintosh; Amiga with at least 256K RAM (joystick optional). The Amiga version was reviewed.

Some games are all strategy, some are all action, but many of the best games require both forethought and quick reflexes. *Skyfox* is one of those hybrid games, and it's clearly one of the best available for the Amiga. With its many levels of difficulty and player options, virtually anyone will find it challenging and rewarding.

The elements of strategy in this game recall the venerable computer game *Star Trek*. You're the last hope of the Federation asteroid base, the only pilot available. What's more, you've got to fly this experimental jet without sufficient training, and you can't even recall everything this advanced craft can do.

The asteroid base has been attacked by The Enemy, one or more immense motherships which convulsively disgorge wave after wave of tanks and planes. *Their* mission is to destroy the Federation Homebase which houses the Skyfox computer and the only place where you can refuel and recharge your shields. From time to time, you must check with your computer's grid map of the entire asteroid to see where enemy forces are massing. If they manage to get close to Homebase, you should try to take them out. If Homebase is destroyed you can still prevail, but it will be far more difficult.

The action elements of the game are among the best you'll ever see: realistic, realtime graphics; excellent stereo sound; complex air and ground battle scenes. Heat-seeking missiles, laser cannons, enemy tanks and planes, clouds, cockpit controls, Homebase, guided missiles, trees, shrubs, and sky are all vivid and believably recreated using computer graphics in three dimensions. *Skyfox* is more than a game; it's an effective visual and aural simulation.

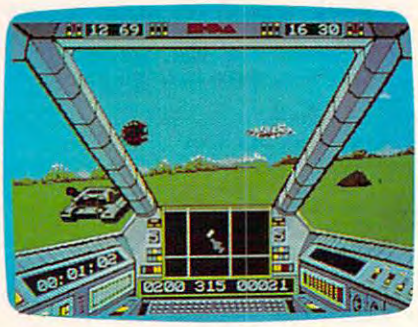
Ace Of The Base

The simulation is made more rich by the large number of options you have during your struggle to overcome The Enemy: a tactical map; zoom maps of individual sectors; automatic pilot; an installation status report; fuel, speed, and shield indicators; x and y coordinates; a compass readout; forward and rear radar scanners; techniques to move between sky and ground battle; and an altitude indicator. Make good use of these tools and you'll find yourself capable of moving up in rank and attempting some of the more drastic invasion scenarios.

Before an invasion starts, you select one of five skill levels ranging from Cadet through Ace of the Base. Then you choose one of the 15 scenarios. There are 7 training scenarios during which you can work to improve the accuracy of your control over the inertial motion of Skyfox and steel your nerves against the smoke and flame and relentless attack of enemy tanks and planes. There are no Motherships during training, so there is a finite number of attackers. Also, Homebase cannot be destroyed.

When you feel confident that you're ready for the real thing, select a Small, Full, or Massive Invasion. These differ primarily in the number of Motherships active during the game. If you eventually become truly skilled, there are the five ultimate invasions during which multiple Motherships attack using different formations and varying strategies to take out your Homebase. These scenarios are called Halo, Alamo, Advancing Wall, Chess, and Cornered.

The Amiga, with its speed, sophisticated graphics, and quality stereo sound, is an excellent medium for this challenging, vividly realized game. The designers and programmers have outdone themselves in exploiting the Amiga's powerful features and have, in Amiga *Skyfox,* created a simulation which rivals the best computer games available in any medium.



Skyfox *is an exciting action-strategy game that reveals much of the potential of the Amiga's graphics and sound.*

Skyfox
Electronic Arts
2755 Campus Drive
San Mateo, CA 94403 $32.95 (64 version)
$39.95 (all other versions)

The Battle Of Antietam

James V. Trunzo

Requirements: Apple II-series computer with at least 48K RAM; Commodore 64 or 128; or an Atari 400/800/XL/XE with at least 48K RAM. Disk only.

Less than a year before the battle of Gettysburg, a Civil War conflict erupted that became known as "the bloodiest day in American history." In Sharpsburg, Maryland, the battle of Antietam produced more than 22,000 casualties, and it has since been one of the most debated encounters of the Civil War.

The Union army, under the command of General McClellan, outnumbered Robert E. Lee's Confederate forces by more than two to one. Yet throughout the course of the battle, the cautious and indecisive McClellan failed to commit the bulk of his army. Along with a number of other blunders, this turned the day's battle into a

All The Best

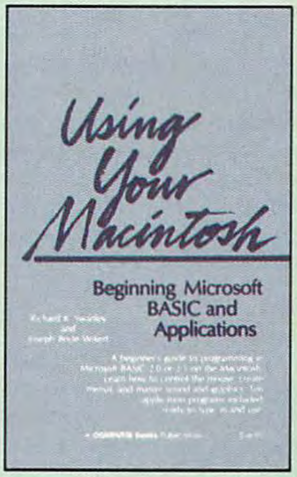
APPLE INFORMATION

Take advantage of the spectacular special features of your Apple II-series and Macintosh computers with these bestsellers from COMPUTE! Books.

Advanced Macintosh BASIC Programming

Philip Calippe, 309 pages
A reference guide and tutorial to Microsoft BASIC which shows you how to use the Macintosh's advanced features to create impressive programs. *A disk is also available which includes programs in the book, $15.95 (0300BDSK).*

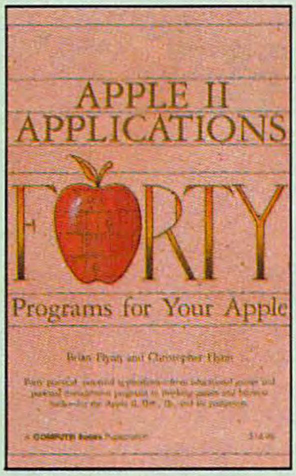
$16.95 ISBN 0-87455-030-0



Using Your Macintosh: Beginning Microsoft BASIC and Applications

Richard K. Swadley and Joseph Boyle Wikert, 274 pages
Necessary and easy-to-understand information about the revolutionary Macintosh along with clear, easy-to-follow explanations of BASIC. Everything from writing your first statement to creating a finished program. *A disk is also available which includes programs in the book, $15.95 (0211BDSK).*

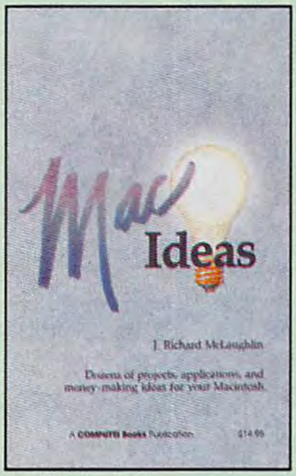
$16.95 ISBN 0-87455-021-1



Apple II Applications: Forty Programs for Your Apple

Brian Flynn and Christopher Flynn, 374 pages
Forty educational and strategy games, business and science applications, and home and personal organizational tools to use on any Apple II-series computer. *A disk package is also available which includes programs in the book. Specify DOS 3.3 or ProDOS, $12.95*

$14.95 ISBN 0-87455-016-5



MacIdeas

J. Richard McLaughlin, 240 pages
More than 100 ways to utilize the Macintosh's powerful graphics capabilities. Beautify everything from personal gifts to correspondence, and learn how to use digitizers to create dazzling graphics.

$14.95 ISBN 0-87455-015-7

Mail this coupon with your payment to COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.
Or call toll-free 800-346-6767 (in NY 212-887-8525).

Please send me the following books and disks:

- ____ **Advanced Macintosh BASIC Programming (030-0), $16.95 each** ____
- ____ **Advanced Macintosh BASIC Programming Disk (0300BDSK), $15.95 each** ____
- ____ **Apple II Applications (016-5), $14.95 each** ____
- ____ **Apple II Applications Disk, DOS 3.3 (0165DSK1), $12.95 each** ____
- ____ **Apple II Applications Disk, ProDOS (0165DSK2), 12.95 each** ____
- ____ **Apple II Applications Book and DOS 3.3 Disk Combination package (050-5), $29.95 each** ____
- ____ **MacIdeas (015-7), $14.95 each** ____
- ____ **Using Your Macintosh (021-1), $16.95 each** ____
- ____ **Using Your Macintosh Disk (0211BDSK), $15.95 each** ____

Subtotal ____
NC residents add 4.5% sales tax ____
Shipping and handling (In U.S. $2.00 per disk or book; airmail $5.00 per item) ____
Total enclosed ____

All orders must be prepaid in U.S. funds.
Payment enclosed (check or money order)
☐ Charge ☐ MasterCard ☐ Visa ☐ American Express

Account No. ____ Exp. Date ____ (required)
Name ____
Address ____
City ____ State ____ Zip ____

Please allow 4–6 weeks for delivery.
Prices subject to change without notice.

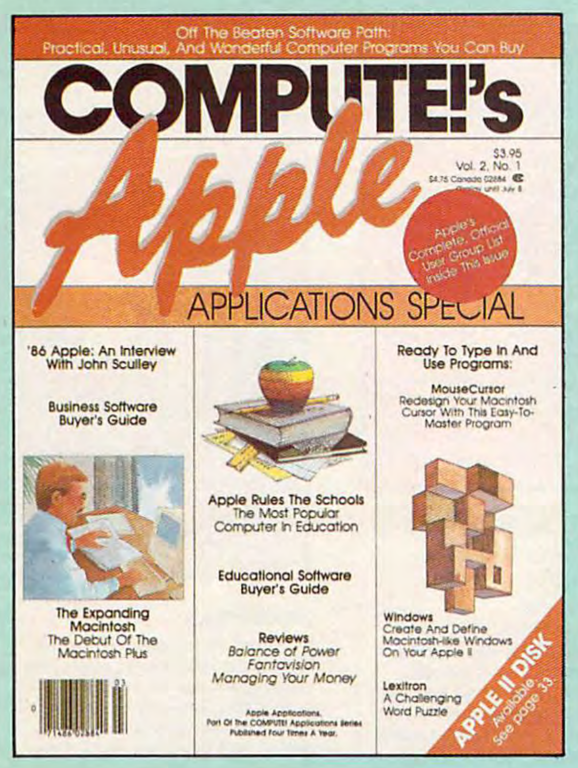
36612211

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

COMPUTE! books are available in the U.K., Europe, the Middle East, and Africa from Holt Saunders, Ltd., 1 St. Anne's Road, Eastbourne, East Sussex BN21 3UN, England, and in Canada from McGraw-Hill, Ryerson Ltd., 330 Progress Ave., Scarborough, Ontario, Canada M1P 2Z5.

COMPUTE!'s *All New* Apple Applications Special

COMPUTE!'s latest *Apple Applications Special* features in-depth articles and interviews, all the inside news about Apple, clearly written tutorials, software buyer's guides, new product information, and valuable ready-to-type-in programs for all Apple users.



Apple owners find these special Apple issues the most understandable, complete, and valuable resources available today.

PLUS

All the programs in *COMPUTE!'s Apple Applications Special* are also available on a timesaving disk, ready to run on your Apple II, II+, IIe, and IIc. The *Disk* costs only $12.95 (plus $2.00 shipping and handling) and gives you immediate access to all the great programs in this special issue.

Look for the Spring/Summer 1986 issue of *COMPUTE!'s Apple Applications Special* on sale where you buy other COMPUTE! publications, or order directly from COMPUTE!. This special issue goes on sale April 8, 1986.

Send in the attached order card or call toll free 800-346-6767 (in NY call 212-887-8525).

Features

- **Business Applications**

'86 Apple: An Interview with John Scully

A wide-ranging interview with the president of Apple. The company's plans for the coming year, its markets, the new Macintosh, and the viability of the Apple II.

Business Software Buyer's Guide

A buyer's guide to the newest Apple II and Macintosh word processors, databases, spreadsheets, and more.

The Expanding Mac

- **Education**

Apple Rules the Schools

Why does Apple have a lock on educational computing? Comments from teachers, administrators, and Apple.

Computers and the Humanities

Educational Software Buyer's Guide

- **The Expanding Apple**

It's New II

A multitude of new hardware and software for the Apple II line—from color printers to Mac-like software—is evaluated.

Weirdware: Off the Beaten Software Path

Weirdware—out of the ordinary software—can turn the Apple II or Macintosh into a telescope, astrological fortune-teller, baby evaluator, and much, much more.

MacAdds: More for the Macintosh

Applications

- **Utilities and Tutorials**

Windows

Create Macintosh-style windows on any Apple II-series computer. Set window size, open, close, and retrieve information.

Mouse Cursor

A Macintosh BASIC utility for altering the mouse pointer. Design data can be saved, then used in other BASIC programs.

Your Personal Ledger

A complete personal financial application for tracking expenses, income, and assets. Easy to use, and packed with features from report generation to customized category codes.

Personal Publishing With Your Macintosh

Tutorial and guide to using such software as *MacPaint* and *MacWrite* to customize letterheads, cards, banners, and more.

Keynote

- **Education and Recreation**

Lexitron

Entertaining word game where players try to beat the clock while finding as many hidden words as possible.

Backgammon

Play the computer in this classic game. This version observes all the rules of standard backgammon.

Apple Automatic Proofreader

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

nightmare encounter and possibly prolonged the Civil War by years. Had McClellan been more aggressive, the Confederacy might have been crushed at Antietam and the course of history changed.

"What might have been" is exactly what makes Strategic Simulation's *The Battle of Antietam* such an excellent game. You can choose to follow the exact order of battle, with troops being committed as they actually were during the real fighting, or you can take total control and have all troops put into action from the start of the battle and attempt to change the outcome of this bloody day in American history.

Like all SSI games, *The Battle of Antietam* has been meticulously researched and is a tactical game on a grand scale, incorporating 17 weapon types plus a wide variety of options. The game can be played on an introductory, intermediate, or advanced level; units may be represented by icons or symbols; units may be hidden or visible; and map details include towns, streams, ridges, and bridges superimposed on a square grid that displays four elevations. There are many other options, as well.

Union Frustration

But it's more than just the accuracy and playability that makes this 11- to 15-hour game so special. Perhaps it's the battle itself.

When using the Activation option, troops are not available to the player until the time at which they historically entered the battle. This creates an extremely realistic simulation. In fact, when I tried commanding the Union forces using this option, I've never experienced such frustration. Turn after turn I watched the valiant blue coats charge the Confederate positions, fighting to gain a bridge or a hill. I watched them dissolve before the Confederate artillery, break ranks, and retreat—while a huge Union force sat dormant within striking range of the enemy. I came away with a much better understanding and appreciation of just what had occurred at Antietam—and this is what a computer simulation is all about.

Beyond these features, *The Battle of Antietam* incorporates such factors as fatigue, chain of command, limbering and unlimbering artillery, mounting and dismounting cavalry, line-of-sight targeting (which requires only a touch of the key to highlight all possible targets), and more tactical control than any other game in its class. The game may be played solitaire—with the computer commanding either force—or two players can compete head-to-head and try to match Lee's genius and avoid McClellan's indecision.

SSI has produced dozens of computer war games, gathering praise from many sources. *The Battle of Antietam*, however, may transcend previous efforts and become a true classic.

The Battle of Antietam
Strategic Simulations, Inc.
883 Stierlin Road
Mountain View, CA 94043-1983
$49.95

OnLine! For Amiga

Philip I. Nelson, Assistant Editor

Requirements: Amiga computer with RS-232C modem.

OnLine! is a full-featured telecommunications program that allows any Amiga to communicate with remote computers, bulletin boards, and commercial information services such as CompuServe. Since *OnLine!* takes full advantage of the Amiga's graphics-oriented operating system, the program is intuitive and convenient to use. In most cases, selecting an option is as simple as moving the mouse pointer to the desired menu item. But don't confuse ease of operation with a lack of features; this program offers a wide range of options, making it suitable for serious applications as well as recreational use.

For most home use (calling an information service, for instance), you'll want to use the default TTY, or dumb terminal configuration. But you can also choose from three popular DEC terminal modes (VT-102, VT-100, and VT-52) or ANSI emulation. The default window—with a status display line at the top, screen borders, and a sizing gadget at the lower-right corner—has room for a 79-column × 22-line text area. Other display options include a borderless 80 × 23 window, which removes the sizing gadget but leaves the status line in place, and a full 80 × 24 window which has neither a status line nor a sizing gadget.

The most novel display feature is the split or *chat* window, which is designed for realtime electronic conferencing (like the CB service on CompuServe). On many terminal programs, realtime conferencing is a very confusing business. Since your own keystrokes are intermixed with incoming characters, it's very difficult to keep track of what you're typing. By echoing only your keystrokes in a separate window, *OnLine!*'s chat feature eliminates the confusion.

Unlike some early Amiga software that completely takes over the machine, *OnLine!* is clearly designed to exist in a multitasking environment. In all configurations except the 80 × 24 window, you can use sizing and/or depth gadgets to gain access to the Workbench or other windows. This welcome feature makes it possible to perform other tasks while the terminal remains active. For instance, you might want to open a new CLI window to check whether a disk has enough space to hold a file that you've captured.

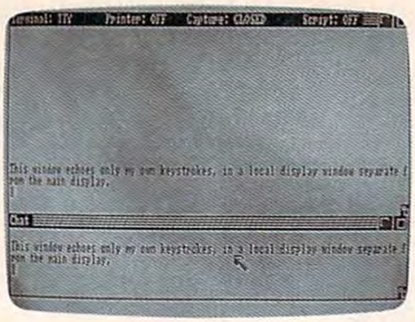
Flexibility

Few things are more frustrating than establishing a communications link only to find that the computer at the other end of the line requires a protocol that your software can't handle. *OnLine!* goes to considerable lengths to provide control over all the parameters you need, without forcing you to specify settings more often than necessary. When you first run the program, it defaults to the configuration used by most commercial information services: 1200 bps (bits per second), 7-bit word length, even parity, and a stop bit of 1. But these parameters (and many more) are easily changed via onscreen menus.

Once you've chosen new settings, you can save them in a *terminal file*, which also includes display choices, phone numbers (for an autodialing modem), and macrokey definitions (see below). Terminal files are a real boon to anyone who calls more than one service regularly. Instead of reconfiguring the program manually each time, you need only set the parameters once for each service and save them in a terminal file. After that, you simply select the desired terminal file from a menu. When *OnLine!* loads the file, it configures the display window, sets all the necessary parameters, and even dials the number for you automatically.

It's easy to see how this sort of automation speeds up and simplifies the process of getting online. Going one step further, you can also customize the way in which the program boots up. Whenever you run *OnLine!*, it looks for a special file named *OnLine!.trm*. If the disk contains a terminal file of that name, the program comes up with the settings specified in the file, and dials the phone number if one is included.

You can also save time by creating a custom *macrokey* definition for one or more of the Amiga's ten function keys. Once a macrokey has been defined, it sends as many as 64 characters to the serial port with only one keypress. In the simplest case, you might program a key to transmit a commonly used command such as BYE or GO AMIGAFORUM. By including control codes and linking together more than one macrokey, it's possible to create much more elaborate one-key sequences.



OnLine! *is a convenient, professional-quality telecommunications program for Amiga computers. This screen shows the chat window feature designed for real-time teleconferencing.*

Unlike some terminal programs, *OnLine!* has no separate phone book as such. Instead, two phone numbers (a primary number and one alternate) can be stored as part of each terminal file. If you need more than two numbers for a certain service, you could store additional numbers in macrokey definitions, which also become part of the terminal file. The autodial feature lets you set the number of times to redial the primary and alternate numbers before giving up. The default number of retries is zero, meaning that if the primary number isn't answered within 30 seconds, *OnLine!* dials the alternate number (if one is supplied) or simply hangs up.

If you've ever had to write a program to transfer data files from one computer to another, you know that character translation, while extremely simple in theory, can soak up a lot of programming time in practice. *OnLine!* lets you edit any of its seven 256-byte character-translation tables (which relate to screen, keyboard, printer, and serial input/output) simply by calling the table from a menu and editing the character values onscreen. This makes it easy to do character translations or filter out undesired characters for various purposes. When streaming input to a printer, for instance, you can check for certain characters which might be interpreted as control codes, producing unwanted results.

Automation

Perhaps the most advanced feature of *OnLine!* is its ability to execute *scripts*. A script file is simply a collection of commands stored in a text file on disk (similar to a batch or script file in AmigaDOS). When you load a script file, *OnLine!* automatically performs all the commands found in the file. In other words, the script feature is actually a mini-language interpreter; you can write simple programs, store them in disk files, and execute them whenever you like. This powerful capability makes it possible for the system to carry out an elaborate series of actions without any supervision on your part.

To illustrate what a script can do, say that you want the program to wait until 3 a.m. (when rates are low), dial up a fictional information service called ChompuSerf, log on to the service, enter Data Library 3 in the area called Amigashop, download a file named EXAMPLE.BAS, log off the service, hang up the phone, and save the captured file to disk. Your script file might look something like this:

```
WAIT UNTIL 03:00
REPLY "ATDT 1 919 555 1212"
WAIT DELAY 50
REPLY "ϒ"
WAIT DELAY 5
REPLY "ϒ"
WAIT STRING "Host:"
REPLY "CIS"
WAIT STRING "User ID:"
REPLY "55555,1212"
WAIT STRING "Password:"
REPLY "BUZZWORD"
WAIT STRING "your choice!"
REPLY "go amigashop"
WAIT STRING ":"
REPLY "DL3"
WAIT STRING ":"
REPLY "DOW"
WAIT STRING ":"
CAPTURE OPEN 100
REPLY "EXAMPLE.BAS"
WAIT STRING ":"
CAPTURE CLOSE
REPLY "BYE"
WAIT DELAY 5
OFFLINE
CAPTURE SAVE "EXAMPLE.BAS"
```

The first command in this script causes *OnLine!* to wait until the system clock equals 03:00, or 3 a.m. (of course, it's your responsibility to set the time correctly at the beginning of the session). The next command calls ChompuSerf by sending a Hayes-format autodial command to the modem. The next two REPLY commands simulate the process of pressing RETURN twice. The following WAIT STRING commands cause the program to pause until a particular character string is received. Each REPLY command sends a character string, so by REPLYing to prompts as needed, we move to the Amigashop section of ChompuSerf, enter Data Library 3, and download the file EXAMPLE.BAS. The CAPTURE OPEN command opens the ASCII capture buffer, specifying a buffer length of

100K. When the capture is complete, we log off ChompuSerf (REPLY "BYE"), hang up the phone (OFFLINE), and save the captured file to disk with CAPTURE SAVE.

The example script is actually quite primitive compared to what *OnLine!*'s command set allows. More advanced commands such as IF, WHEN, ASK, JUMP, SKIP, and ABORT permit the script to test for certain conditions, branch to other parts of the script program, and interact with the user to a certain extent. The DO command even lets you load and execute a second script file from within the first.

Writing an automated script like the example shown here requires that you know in advance exactly what the remote system will send in the way of prompts and what you must supply as responses. The simplest way to glean such information is to note each prompt/reply sequence on paper as you go through a typical session. Once that's done, you can write the script file using the ED system editor or a word processor.

But that process takes time and multiplies the chance for errors. *OnLine*'s *learn mode* automates the process of creating script files by letting you capture the relevant information on the fly. In learn mode, the program automatically records the most recent prompt as well as your last reply, giving you a chance to edit each string on the spot and insert additional commands before adding it to the script file. At the end of a session, you should have a script that requires little or no extra editing.

Transfer Options

OnLine! offers several options for capturing or sending data files, including ASCII capture, standard XMODEM protocol, XMODEM with CRC (*cyclic redundancy checksum*) error-checking, and HVP (Hayes verification protocol). The timing requirements for standard XMODEM are relaxed somewhat to facilitate communications via packet-switching networks. Though it's not supported by every information service, CRC error-checking improves the reliability of XMODEM transfers.

One headache that confronts Amiga owners concerns XMODEM transfers of executable machine language files. Since the XMODEM protocol always sends a file in even 128-byte chunks, any file that doesn't divide evenly by 128 is padded with extra characters when you download it with XMODEM. If you try to load and run a padded file, AmigaDOS notices the padding, concludes that the file is not executable, and refuses to run it. Chopping off the padding is a simple matter from BASIC, but the file is useless until that's done. So this problem adds just one more layer of aggravation and delay to the process of getting someone else's program to work on your computer.

It's worth noting that the padding problem applies only to XMODEM transfers—more specifically, to XMODEM transfers of executable machine language files or other binary files for which exact file length is critical. It shouldn't affect text that you save from the capture buffer, or ASCII text files (including BASIC programs in ASCII form) downloaded with XMODEM. Of course, the padding problem isn't unique to *OnLine!* or any other terminal program. It's a consequence of the way that XMODEM and AmigaDOS treat certain files, and occurs with any Amiga terminal program that supports XMODEM.

OnLine! does not contain any feature to help you chop executable files downloaded with XMODEM. However, it does support HVP protocol (compatible with *Smartcom*) which can transfer executable files without padding. The only problem with HVP, or any protocol other than standard XMODEM, is that not everyone uses it. (Perhaps the best solution is for programmers to pad their executable files before uploading them to public bulletin boards.)

Confusing Manual

While the *OnLine!* instruction manual is fairly complete, it is disorganized. All the information is there—someplace—but it's not always easy to find. Despite the manual's length of 100 pages, there is no index. Fortunately, documentation is less important for a menu-oriented program of this type, which displays nearly every option onscreen. Many people will be able to use *OnLine!* without glancing at the manual. But some important program features—learn mode, for instance—don't appear in the menus at all.

On the whole, however, *OnLine!* is a very impressive package with the look and feel of a finished, professional product. It's convenient, reliable, and well-integrated with the Amiga's personality. Another plus is the quality of customer service. The authors (Micro-Systems Software, Inc.) offer technical support in two different forms: on voice lines during regular business hours, and on their own 24-hour, 7-day BBS. I found that questions to the customer BBS were answered very promptly.

OnLine!
Micro-Systems Software, Inc.
4301-18 Oak Circle
Boca Raton, FL 33431
$69.95

Hippo Computer Almanac For Atari ST

George Miller
Assistant Technical Editor

Requirements: Atari ST computer with at least one disk drive. Printer optional.

Do you know how many ounces are in a liter? Quick, what time is it in Moscow? What's the zip code for Denver? Who won the Super Bowl in 1974?

No, we're not playing another version of Trivial Pursuit. These are questions you can answer in seconds with the *Hippo Computer Almanac* for the Atari ST, a valuable information resource that points the way toward a new generation of intelligent software.

The higher processing speeds and greater disk capacities available with the new generation of personal computers are making possible more powerful and sophisticated programs. For example, much larger databases are becoming available for use in the home. Although the *Hippo Computer Almanac* is not yet in the class of an encyclopedia on a CD-ROM, it is loaded with information. Over 35,000 pieces of information, in fact, according to Hippo.

It Understands English

Like any good almanac, this electronic repository contains information on such general topics as history, geography, sports, languages, science, awards, and units of measure.

Perhaps the best feature of all is that you communicate with the program by typing plain English sentences. A parser routine swiftly evaluates your query, and the program usually retrieves the information in less than ten seconds. If the almanac doesn't know the answer to a question, there's no cryptic comment or error message. The screen simply displays, "I don't know."

Of course, even with a first-rate parser, there are always going to be occasions when the program won't follow your questions. However, the *Almanac* does have the ability to find the closest match to any request, and it tries to satisfy any query.

If, after several attempts, you still can't make the program understand your question, just type HELP. Online help is always available in all categories. The help screens are easy to understand and even offer sample questions illustrating the format for communicating with the program. As your familiarity with the *Almanac* increases, you'll learn how to communicate in the least number of words. For instance, "Time London" yields the time of day in London, England, eliminating the need to type "What time is it in London, England?"

A Personalized Almanac

You can also customize your version of the *Almanac*. For instance, it's easy to set up the database so the program knows where you are geographically. This makes it possible for the *Almanac* to calculate time zone differences and mileages between your home town and distant lands. You can also use the "remember" command to store important personal information in the *Almanac*, such as birthdays, anniversaries, and phone numbers.

The *Almanac* is easy to use without extensive instructions. In fact, a single information sheet is provided instead of a manual. There is also an easy-to-use print option that lets you make hardcopies of anything you call up.

Browsing through the *Hippo Computer Almanac* is fun. It's an engaging program that entertains at the same time that it offers a useful database of information.

Hippo Computer Almanac
Hippopotamus Software, Inc.
985 University Avenue, Suite 12
Los Gatos, CA 95030
$34.95

Zoomracks For Atari ST

Arthur Leyenberger

Requirements: Atari ST computer with disk drive. Printer optional.

Zoomracks by Quickview Systems is a powerful, easy-to-use database manager that lets you keep track of lists, names, addresses, notes, schedules—almost anything you can think of—in a unique and interesting way. What's unique about the program is the concept of the "rack."

Consider a familiar timecard rack—the vertical holder that sits next to the time clock and holds employee timecards. The first line of each card is always visible. You can remove any card to examine its contents. Cards can also be inserted or moved into other slots in the same or adjoining racks. Cards in the racks are typically in the same form (timecards), but contain different information, such as names and hours worked. They may be arranged in some order, such as by name or employee number.

This describes the visual metaphor upon which *Zoomracks* is based—the card rack. It is a familiar concept and translates well to the computer. When you choose a card from the rack in order to see its contents, you notice that it has several fields, each of which shows the top line of information just as all the cards appear. Each field can be pulled up to expose as much as three pages of information.

Stretching The Rack

The way your information is organized is always visually obvious because the screen shows as many as ten racks at once. The number of cards in each rack is limited only by the amount of computer memory, and the racks grow or shrink as required. If your rack is too large to fit on the screen, it can be scrolled. Or you can search for the card or field you're looking for. The cards in one rack can be sorted by any field, and each card can have up to 29 fields.

Zoomracks offers three different field types: short fields, text fields, and columns. Short fields are similar to those found in traditional database programs. One field is displayed at a time on each line. Text fields are used for multiple notes. The document (your notes) is displayed across the entire width of the screen on consecutive lines. Finally, the column field is used for spreadsheet-type information—for example, sales orders.

Rack formats can be inserted and deleted by moving fields; cards and fields can be copied between racks; you can do simple word processing, since any field can be up to 250 lines long; and the cards and racks can be printed in many different formats.

One interesting feature of *Zoomracks* is its macro capability. Macros are any series of *Zoomracks* commands that are strung together and issued at once by a single keypress. You can have one rack with up to 26 macros. There are several sample macros provided, and

Zoomracks *uses a unique visual metaphor to let you organize and retrieve information.*

one serves as a tutorial for the program.

Mail-Merge Feature

The program disk contains several sample racks as well. One sample which is useful for more than just learning about *Zoomracks* is a mail-merge template. One card within a rack serves as a form letter, and the card-merge macro can be used to print out a rack of cards consisting of names, addresses, and salutations. With a little imagination, you can develop all kinds of applications by using macros.

All in all, *Zoomracks* is a practical and even a fun way to keep track of various types of information. With its visual interface and zooming feature, you can always visualize your data as you want—from a broad overview of the whole database to a specific detail of a single field. Help is available at any time and the menus are straightforward. As you get more experienced, you can use the commands instead of the menus. In either case, there are few rules to follow and few limitations.

Zoomracks is a well-done program and a unique concept. If you need a database manager and want to get up to speed as quickly as possible, *Zoomracks* is an excellent choice.

Zoomracks
Quickview Systems
146 Main Street
Los Altos, CA 94022
$80

Stickybear Learning Games For Apple And Commodore

Karen G. McCullough

Requirements: Apple II-series computer with at least 48K RAM and a disk drive. Joystick optional. Commodore 64 version scheduled for release by this summer.

With their Stickybear series, Optimum Resources and Weekly Reader Software have developed a reputation for producing software that is reliable, educational, and entertaining. They maintain those high standards with three new releases: *Stickybear Typing, Stickybear Town Builder,* and *Stickybear Spellgrabber.*

Typing is an application ideally suited to computerized instruction—it's an area where the computer can do a better job than traditional methods of teaching. A good typing tutor program provides immediate feedback—both aural and visual—for incorrect keypresses, and allows a student to progress automatically through levels as each is mastered, rather than dictating progress with a schedule or lesson plan.

Stickybear Typing does all this and more. Each of the program's 30 levels introduces the student to the keys covered in the lesson, then offers practice using them. The lower half of the screen displays the keyboard; as keys are highlighted one at a time, the student must press the corresponding key on the computer's keyboard. A correct keypress prints the letter at the top of the screen. Incorrect keypresses make a low "bloop" sound, and the letter doesn't appear. At the end of each two screens of typing practice, the student gets a progress report which shows the starting level, current level, number of words typed per minute, number of errors, and corrected words per minute.

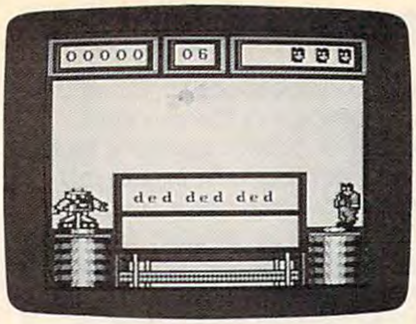
A Typing Game

Another section of the program—Stickybear Thump—allows typing practice in the form of a game. Stickybear and a robot throw things at each other while the player copies lines of letters displayed on the screen. The robot throws boxes at preset intervals; each time a line is completed, Stickybear throws a ball at the robot. The faster you type, the more balls Stickybear throws, the more points you get, and so on.

A third section of *Stickybear Typing,* the Stickybear Stories Module, provides typing practice of a more practical sort—copying amusing stories, paragraphs, and jokes.

Stickybear Typing has a number of nice features. Up to 25 names can be stored on the disk with current level information for each person. The sound can be toggled on and off, as can a hands display which illustrates proper finger placement on the keyboard. In two sections of the program, you can choose either typewriter mode (you must press RETURN at the end of each line, and you can't backspace to that line) or word processing mode (freestyle typing).

Although *Stickybear Typing* is intended primarily for children, it can be used by adults just as effectively. We found only one problem with the program: A decent typist can outrun it. Particularly in the game sections, frustrating errors can occur as the program drops letters which are typed too quickly. However, most students won't be fast enough to experience that problem, at least at first.



Stickybear Typing *offers several ways for youngsters to sharpen their keyboard skills (Apple version).*

Build A Town

Stickybear Town Builder, for children ages six to nine, lets the youngsters build their own towns on the screen, drive through them with a small keyboard- or joystick-controlled car, hunt for hidden keys, and learn some elementary map-reading skills in the process. Towns can be saved and loaded again later, or you can use one of three towns provided on the disk. The graphics are attractive, and the program is easy enough to be used by children even younger than six. But children at the older end of the suggested age range may not find the program challenging enough to hold their attention for long.

If your child needs work on spelling, *Stickybear Spellgrabber* might be the answer. Three different games help a child learn selected word lists. All three games are fun, challenging, and really can help with spelling drills. A nice feature of the program allows you to enter your own spelling list or use one of the four lists included (keyed to grades 1–4). Stickybear can be controlled with either keyboard or joystick. While the joystick is slightly easier to use, both require practice to master. Unlike *Town Builder,* all three games are difficult enough to be challenging even to nine- or ten-year-olds, as well as educational.

Stickybear Typing
Stickybear Town Builder
Stickybear Spellgrabber
Weekly Reader Family Software
245 Long Hill Road
Middletown, CT 06457
$39.95 each (Apple versions)
$29.95 each (64 versions)

Kennedy Approach
For Commodore And Atari

David and Robin Minnick

Requirements: Commodore 64 or 128 (in 64 mode); or an Atari 400/800, XL, or XE with at least 48K RAM. Disk drive and joystick also required. The Commodore version was reviewed.

It's 10:53 a.m.

You're in the midst of your second shift as an air traffic controller. Six flights await your clearance for takeoff. Five more are waiting to land. Compounding your headache are a thunderstorm approaching from the west and the Concorde approaching from the east.

Suddenly you hear, "This is United 101. Emergency! Eight minutes fuel!"

The Concorde moves at eight miles every minute. Within two minutes the planes will be at a point of intersection. Unless United 101 gets on the ground fast, lives will be lost.

Your palms begin to sweat.

"United 101. Turn left, heading 90 degrees. Descend to 3,000 feet. Air France 314. Hold right at VDR at 5000 feet."

Oh no! you think, staring at the screen. *I forgot Delta 626 coming in at the same altitude!*

The conflict buzzer sounds.

Your spouse looks up from the couch. "Could you please turn that thing down?"

It's Just A Simulation

This is *Kennedy Approach,* an air traffic control simulation from Micro Prose. It puts you in the seat of an air traffic controller in one of five U.S. cities. Each airport presents you with skill levels ranging from 1 (Atlanta—a challenging beginning) to 5 (New York City—no margin for error).

In *Kennedy Approach,* you work a shift of approximately ten minutes realtime, longer at the higher levels. At the end of your shift, your performance is evaluated and you're promoted, given a bonus, or fired. Additional options let you continue your career, see an instant replay, save your shift to resume playing later, or return to the main screen.

It's only a simulation, a game, you tell yourself between shifts—but the sweat on your palms when you play *Kennedy Approach* is quite real.

Keyboard or joystick controls are used to establish contact with a plane. Then the joystick is used to change its heading and/or altitude. A push of the fire button prompts an exchange of dia-

Keeping the friendly skies friendly is a frenzied job in Kennedy Approach, *an air traffic control simulation (Commodore 64 version).*

log between you and the pilot. Probably the most delightful feature of the program is the use of digitized voices for this exchange. This is software-driven speech synthesis from Electronics Speech Systems. The dialogs have the quality of genuine "black box" air traffic recordings.

The graphics overall are very good, particularly the thunderstorms, but a few effects require getting used to. The one representing a plane's location is somewhat confusing, and it's difficult at first to decipher the display of flight plans. Both these problems are conquered by familiarity.

Some Minor Quirks

There are a few quirks in *Kennedy Approach.* Planes start to wrap around the screen, a sight which can be disconcerting to the newly hired controller. Routing flights into a holding pattern is a lipbiting maneuver, as this requires you to press the fire button at the right moment while commands are sequentially displayed in the command line. This is the most difficult task in the program, and it seems that it could be accomplished more easily.

Another oversight is that *Kennedy Approach* lacks a disk directory function for selecting which shift to retrieve.

The instruction manual is superb in providing information about the air traffic control aspects of the simulation. This technical information allows even the beginner to feel familiar with the new environment. One small flaw, though: At one point the manual directs you to a nonexistent Section VI, leaving you to your ingenuity and experience to discover how to instruct the pilot to climb to the desired altitude at takeoff. (This is corrected in later editions of the manual. Users with early manuals should refer to B-3 instead of Section VI.)

Despite these small problems—they're the only ones we found and are minor compared to the whole package—*Kennedy Approach* is a fascinating, well-designed simulation for someone who wants to get a taste of what air traffic controllers do all day (and night). More simulation than game, it still elicits game-type responses. If you judge a game by how it affects your psyche, by how excited you get, and by how nervous it makes you, *Kennedy Approach* gets a clammy hands rating of 9 out of a possible 10.

Kennedy Approach
Micro Prose Software
120 Lakefront Drive
Hunt Valley, MD 21030
$34.95 ©

Sideways Text For Atari

Bill Morris

Here's a short machine language routine that converts your lowercase letters (a–z) to uppercase sideways letters. Why? Well, it's so short that it's worth typing in just to see the amusing effect, but it's also useful for labeling charts and designing one-of-a-kind title screens. The program works on any Atari 400/800, XL, or XE.

Wouldn't it be nice to have sideways letters that could be displayed anywhere on a GRAPHICS 0 screen? Imagine the interesting title displays you could add to programs. Or, from a more practical standpoint, sideways letters could be more than just a show-off effect for charting programs—they could become a necessity.

One way to get sideways letters is to spend a couple of hours with graph paper or a character editor to redefine the lowercase character set. But that would be the hard way. Such a laborious task is best left to a labor-saving device such as your Atari computer.

The program below contains a machine language routine that decides where in memory to place the new character set, relocates the set to that area, changes the character base pointer, erases the lowercase alphabet, and replaces it with uppercase letters that are rotated 90 degrees to the left.

You might notice that the machine language routine doesn't contain any data to define what the sideways letters should look like. Instead, it actually flips each letter mathematically before relocating it in memory. It does all this in about one second and takes up less space in your BASIC program than would the DATA statements alone if you were doing it the hard way.

Sideways Text In Action

If you want to see sideways text on your own computer screen, just type in the program, save a copy on disk or tape, and then run it. What you'll see is the word SIDEWAYS displayed in GRAPHICS 0 actually turned sideways. Next to this you'll see the word TEXT in normal letters.

Everything appears on a light screen with dark characters. On the normal default screen of white letters on a blue background, the sideways text can be hard to read, so dark letters are preferable. Also, for charts, you might want to blank out the screen borders by adding this line:

95 POKE 712,PEEK(710)

Lines 40–90 POKE the machine language routine into memory page 6, but once the routine is executed, you can reuse this memory for some other purpose without affecting the sideways text. It stays sideways until you press SYSTEM RESET.

Sideways Text For Atari

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
IK 10 ? CHR$(125)
JA 20 POKE 559,0
NF 30 GOSUB 20000
MJ 40 POKE 559,34
BE 50 X=USR(1536)
FM 60 POKE 710,158:POKE 709,
      0
FN 70 ? "  s"
GE 80 ? "  y"
EN 90 ? "  a"
NA 100 ? "  w TEXT"
HK 110 ? "  e"
HK 120 ? "  d"
IA 130 ? "  i"
IL 140 ? "  s"
BJ 150 ? "  "
GO 160 END
BL 20000 FOR A=1536 TO 1715:
         READ B:POKE A,B:NEX
         T A:RETURN
MC 20010 DATA 104,165,89,56,
         233,4
BK 20020 DATA 141,244,2,133,
         205,169
BK 20030 DATA 224,133,207,16
         9,0,133
BB 20040 DATA 204,133,206,16
         2,0,160
BG 20050 DATA 0,177,206,145,
         204,200
IB 20060 DATA 208,249,230,20
         5,230,207
OO 20070 DATA 232,224,4,208,
         238,32
FL 20080 DATA 167,6,160,0,16
         9,0
IC 20090 DATA 145,204,200,19
         2,216,208
JB 20100 DATA 249,32,167,6,1
         69,0
LM 20110 DATA 141,186,6,141,
         182,6
MB 20120 DATA 141,183,6,169,
         8,141
MF 20130 DATA 184,6,141,185,
         6,174
CJ 20140 DATA 182,6,169,128,
         141,181
GB 20150 DATA 6,172,183,6,16
         9,1
LP 20160 DATA 141,180,6,189,
         0,225
IO 20170 DATA 45,180,6,205,1
         80,6
PI 20180 DATA 208,8,177,204,
         24,109
CG 20190 DATA 181,6,145,204,
         173,180
HI 20200 DATA 6,10,141,180,6
         ,200
CC 20210 DATA 204,185,6,208,
         224,173
IN 20220 DATA 181,6,74,141,1
         81,6
CB 20230 DATA 232,236,184,6,
         208,203
MD 20240 DATA 173,184,6,141,
         182,6
IL 20250 DATA 141,183,6,24,1
         05,8
MD 20260 DATA 141,184,6,141,
         185,6
ND 20270 DATA 238,186,6,173,
         186,6
DA 20280 DATA 201,27,208,167
         ,96,173
IF 20290 DATA 244,2,24,105,3
         ,133
OO 20300 DATA 205,169,0,133,
         204,96
```

©

Loading And Linking Commodore Programs

Part 4: Overlaying

Jim Butterfield, Associate Editor

This installment of Jim Butterfield's series on loading and linking Commodore programs talks about overlays—a technique that allows a program to call in additional subroutines and other data. The principles apply to most Commodore computers, including the 64, 128, VIC-20, PET, Plus/4, and 16.

There are three major ways of connecting Commodore programs together. *Chaining* allows several programs to perform a job, each program continuing the work that a previous program began. *Linking* enables one program to call up another, with the new program starting fresh on a new task. *Overlaying* allows a main program to call in supplementary material such as machine language subroutines, data tables, or display information. This article discusses overlay programming techniques. (Though the example programs are designed for a disk drive, you should be able to change most of them to work with tape by replacing ,8,1 with ,1,1.)

In some situations a computer program may need extra pieces of information to perform its task. The extra material may be one or more programs (often machine language), or it could be pure data. Data can be of several types: information, display screens, character sets, sprite shapes, or whatever. The difference between overlaying and chaining or load linking is that the main program stays in memory at all times, calling up the modules it needs.

Why Overlay?

The classic reason to overlay programs is so that a main program can call up a machine language module to do a specific job. This permits you to keep a library of special programs on disk and call in each program as it is needed. For example, you might bring in one machine language program to scan through a file, searching for information; another might be used to display data neatly on the screen; yet another module could be called to handle printer output, and so on. In the simplest case, only one program module is brought in at a time, and a certain section of the computer's memory is set aside to hold the current program. This lets you run programs which are, in effect, much larger than the amount of memory in your computer.

One obvious use for this technique is to bring in a series of attractive high-resolution graphics screens. Since each hi-res screen requires 8,000 bytes of memory (with more needed for color information), it's not practical to keep more than one or two in memory at a time. But a disk can hold the data for many hi-res screens. By calling in each screen only when it's needed, you can display dozens of hi-res pictures in the course of a program run.

The same factors apply to other sorts of data, too. For instance, a program could use many different sprite shapes as it runs. Sprite-animated figures could change from bicycles to cars, and later to horses, elephants, or boats as a schedule of race events progresses. All that's required is to replace one set of sprite shapes with a new set by means of overlaying.

Alternate character sets also require extensive amounts of data, usually thousands of bytes for each different set. If you want to switch from Roman (the characters you're reading right now) to Greek, Arabic, Hebrew, Russian, or whatever, simply haul in each new character set as you need it.

Breaking The Chain

Before you overlay information, you must set aside space to hold it. This isn't a new requirement: Regardless of where the data comes from, it's always necessary to allocate room for sprite shapes, hi-res screens, machine language programs, and so on. So we won't repeat the familiar methods of setting aside memory for such purposes.

Let's work through the sequence of events that occur when you bring in an overlay module. Keep in mind that the BASIC program itself is not replaced—the program is still present and running.

The first step is for the BASIC program to load the desired module with a command like LOAD "MODULE",8,1. (The ,1 at the end of the LOAD command is needed on most Commodore computers to specify a *nonrelocating* load—one that loads the file back into the exact part of memory from which it was saved.)

Here comes the tricky part. When the load is complete, the computer thinks that it has performed a *chain*. It concludes (wrongly in this case) that the old BASIC program has been replaced by a new one. None of the program's existing variables are erased or changed, but *the computer reruns the BASIC program from its first line.*

This phenomenon isn't a bug; it's simply what the designers intended to happen whenever you LOAD from within a BASIC program. However, it raises a puzzling problem for beginners. If you write a program that begins with the line 10 LOAD "MODULE",8,1 and run it, here's what happens. The MODULE file is loaded. Then the program reruns, beginning at line 10. So MODULE is loaded again. Then the program reruns again, loading MODULE again, which causes another restart, and so on. Until you press RUN/STOP, the program continues forever.

Fortunately, there's an easy solution. Because LOAD from within a program doesn't destroy existing variables, we can change a variable when the load occurs and use it to branch around the LOAD command when the program restarts. It's like building a bypass around the LOAD after the overlay is complete. Take a look at this program fragment:

```
10 IF A=1 GOTO 40
20 A=1
30 LOAD "MODULE",8,1
40 REM PROGRAM CONTINUES...
```

Let's trace what happens when this program runs. The first time it's run, the variable A is equal to 0 (it hasn't been defined yet). So the IF test in line 10 (which tests for the condition A=1) fails, and we don't branch to line 40. Instead, the program proceeds to the next line. Line 20 then makes A equal to 1. Line 30 loads the MODULE file to wherever it's going in memory. At this point (the end of line 30), the program goes back to the first statement. This time the IF test is true (A *is* equal to 1), so we branch to line 40. The program continues without getting caught in an endless series of loads. You could also condense the whole operation into one program line:

```
10 IF A=0 THEN A=1: LOAD "MODULE",8,1
20 REM PROGRAM CONTINUES
```

This example combines the IF test, the setting of A to 1, and the LOAD command all in one line. Another option is to replace line 10 of the original example with 10 ON A GOTO 40. In a moment, we'll use a variation of this technique to allow for several overlays.

Setting Up Files

Let's write an example geared to the Commodore 64. We'll overlay three items: a graphics screen and two small machine language programs. The screen will load into the usual screen memory area, locations 1024–2023. The machine language programs will come into the cassette buffer, which starts at location 828 on the 64. (Because this example uses the cassette buffer, it works only with disk.) Only one machine language module will be in memory at a time.

Enter NEW, then type in this program. It creates a screen that will be loaded later.

```
100 DATA 8,1,16,16,25,32,2,
    9,18,20,8,4,1,25
110 OPEN 1,8,2,"0:SCREEN,P,
    W"
120 PRINT#1,CHR$(0);CHR$(4)
    ;
130 FOR J=1 TO 986
140 PRINT#1,CHR$(32);
150 IF J<>494 GOTO 200
160 FOR K=1 TO 14
170 READ X
180 PRINT#1,CHR$(X);
190 NEXT K
200 NEXT J
210 CLOSE 1
```

Make sure that lines 120, 140, and 180 each end with a semicolon. When you run this program, it creates a file called SCREEN which is four disk blocks in length. When that's done, enter NEW again and type in the next program. This one creates a machine language program called MLA. When the ML program loads into memory, it will do three small jobs: It will clear the screen, change the screen background color to white, and set the screen's POKE color to red.

```
100 DATA 60,3
110 DATA 169,147,32,210,255
120 DATA 169,31,32,210,255
130 DATA 169,1,141,33,208
140 DATA 169,0,133,252,169,
    216,133,253
150 DATA 162,4,169,2,160,0
160 DATA 145,252,200,208,25
    1
170 DATA 230,253,202,208,24
    6,96
200 A=42
210 FOR J=1 TO A
220 READ X
230 T=T+X
240 NEXT J
250 IF T<>6238 THEN STOP
260 RESTORE
270 OPEN 8,8,8,"0:MLA,P,W"
280 FOR J=1 TO A
290 READ X
300 PRINT#8,CHR$(X);
310 NEXT J
320 CLOSE 8
```

Be sure that line 300 ends with a semicolon. Run the program; if it stops at line 250, you have an error in one of the DATA statements.

Once that's done, enter NEW again. The next generator program creates a machine language routine to blink the screen. This ML module, which we'll call MLB, will occupy the same part of memory as MLA. The memory conflict isn't important since we'll load the programs one at a time. Type in and run this program:

```
100 DATA 60,3
110 DATA 169,0,133,252,173,
    136,2
120 DATA 133,253,162,4,160,
    0
130 DATA 177,252,201,32,240
    ,4
140 DATA 73,1?28,145,252,200
    ,208,243
150 DATA 230,253,202,208,23
    8,96
200 A=34
210 FOR J=1 TO A
220 READ X
230 T=T+X
240 NEXT J
250 IF T<>5022 THEN STOP
260 RESTORE
270 OPEN 8,8,8,"0:MLB,P,W"
280 FOR J=1 TO A
290 READ X
```

```
300 PRINT#8,CHR$(X);
310 NEXT J
320 CLOSE 8
```

Be sure to put a semicolon at the end of line 300. If you've typed the program correctly, it writes the ML program MLB to disk. At this point, all of the modules are complete. Let's write the main program to tie it all together.

The Main Program

Enter NEW and type in the following program lines. We'll start with a line that dispatches the program to the correct line after each load:

```
100 ON X GOTO 130,160,180
```

The first load brings in the machine language program MLA.

```
110 X=1
120 LOAD "0:MLA",8,1
```

After the first load is complete, line 100 sends us to line 130, where we activate the ML program with SYS:

```
130 SYS 828
```

The next two lines bring in the graphics screen.

```
140 X=2
150 LOAD "0:SCREEN",8,1
```

When the screen has loaded, you'll see the message it contains. After the second load is done, line 100 sends us to line 160, where we bring in the second machine language program:

```
160 X=3
170 LOAD "0:MLB",8,1
```

We resume at line 180 (courtesy of line 100) with a screen in place, the colors set as desired, and a blink program waiting to be called with another SYS command. Let's finish off with a loop to flash the message.

```
180 FOR J=1 TO 20
190 SYS 828
200 FOR K=1 TO 100
210 NEXT K
220 NEXT J
```

That's all it takes. It's a simple example, but the program shows the potential of the overlay technique.

Self-Chaining

Earlier in this series, we mentioned *self-chaining*, a method of restarting a program that has snarled itself inside several levels of subroutines. Again, keep in mind that prevention is the best way to avoid this problem. Good program structure should ensure that you never get tangled up in your own code. But occasionally you may program yourself into a corner and need a simple way to get out.

Assuming that you've gotten into this deplorable situation somehow, you can escape by making the dubious program chain to itself. The chaining activity cancels all FOR-NEXT loops and subroutine RETURNs, and also RESTOREs the DATA pointer to the very first DATA statement in the program. However, all existing variables are preserved, and all open files (if any) remain open.

Don't misunderstand what a self-chain does. The program text itself doesn't change—all you've done is reload the same program lines into memory. But the act of doing so untangled the snarled subroutines and FOR-NEXT loops and restarted the program from its first line. Other than that, everything remains as it was before the self-chain.

Since it's the chaining (not the loading) that does the trick, we can skip loading the program itself. Instead, we can overlay a single byte somewhere in memory to trigger the chaining process. To illustrate, let's write to disk a simple one-byte program file that will load the useless byte to some unimportant memory location. The chaining action that accompanies the load will do the job we want.

To write this file, type NEW and enter the following program:

```
100 DATA 255,0,0
270 OPEN 8,8,8,"0:DUMMY,P,W
"
280 FOR J=1 TO 3
290 READ X
300 PRINT#8,CHR$(X);
310 NEXT J
320 CLOSE 8
```

Again, be sure that there is a semicolon at the end of line 300. When you run this program, it creates a tiny file named DUMMY. Now let's repeat the dreadful program that we used before. Again, please don't write programs this way; it's here just to illustrate the point. Type NEW and enter this program:

```
100 IF N>0 GOTO 130
110 PRINT "NAME LIST"
120 DIM N$(50)
130 PRINT
140 PRINT "DO YOU WANT TO -
-"
150 PRINT "1. ENTER NAMES"
160 PRINT "2. LIST NAMES"
170 PRINT "3. QUIT"
180 INPUT "YOUR CHOICE";C
190 ON C GOSUB 210,310,350
200 GOTO 130
210 PRINT "ENTER EACH NAME"
220 PRINT "FOLLOWED BY AN '
*' CHARACTER"
230 PRINT "TO END ENTRY"
240 GOSUB 260
250 GOTO 240
260 INPUT N$
270 IF N$="*" OR N=50 THEN
LOAD "DUMMY",8
280 N=N+1
290 N$(N)=N$
300 RETURN
310 FOR J=1 TO N
320 PRINT N$(J)
330 NEXT J
340 RETURN
350 END
```

Try to write programs in such a way that you don't get into the problem shown above. By the time the program reaches line 210, it's in a subroutine. At line 260, it's nested within a second subroutine. When line 270 discovers that an exit is wanted, we're almost stuck and don't dare GOTO 130, which would leave unRETURNed subroutine addresses on the computer's internal stack.

Here's how to escape. At line 270, LOAD the one-byte DUMMY file. The load does nothing, but the act of chaining untangles the rest of the mess. How does this compare to our first solution of the same problem, where the entire program chained to itself? You get the same results, but gain in speed because you're loading a much smaller file.

Overlaying, like the other methods examined in this series, becomes especially useful in big-program situations, and generally eases the burden of bringing large amounts of data into memory when it's needed. The computer still thinks that it's performing a chain, but overlaying uses the same general technique for a different purpose. Once you understand the difference between chaining and overlaying, you can write even more powerful, flexible programs. ©

Custom Title Bars For ST BASIC

George Miller, Assistant Technical Editor

This short program demonstrates how to put a custom title on ST BASIC's Output window. It works on all Atari ST-series computers.

ST BASIC puts four windows on the screen entitled Command, List, Edit, and Output. The Output window is where your programs actually run, and the window always displays the same title at the top of the screen: Output. By now you're probably tired of staring at this title bar and wish there was some way to change it.

Fortunately, there is. Not with a built-in BASIC command, however. You have to call a routine in a part of the ST's operating system known as AES (Application Environment Services). The job is not difficult, but the ST BASIC manual lacks the necessary information for making system calls.

When programming the ST, it's helpful to remember that the operating system contains many routines which can be of help. These routines are part of GEM, the Graphics Environment Manager, which is divided into two sections: AES and the VDI (Virtual Device Interface). These libraries contain almost all the routines necessary to handle screen output. Although VDI and AES routines are most easily accessed by programmers using C or machine language, ST BASIC programmers can also call them with the VDISYS and GEMSYS commands. It requires a little extra effort, though.

The short routine listed below, "Custom Title Bars," is an example of a GEMSYS call to the AES library. It can be inserted into any ST BASIC program to display your program's title on the Output window's title bar. Run the routine to see what it does; then modify it in the following ways when using it in your own programs:

1. Change line 20 to assign to the string variable *title$* the name to be displayed in the title bar.

2. Delete line 40, the END statement, and insert your own program at this point. However, be sure you insert an END statement at the end of your program and before line 63000. Otherwise, your program will fall through into the subroutine and cause an error.

Before actually making the GEMSYS call in line 63040, the routine POKEs several parameters into system variables at the addresses pointed to by the built-in BASIC variable *gintin*. These parameters are required by this AES routine. The setup is done in lines 63010–63040.

More information about calling VDI and AES routines can be found in the Atari documentation available to software developers and in *COMPUTE!'s ST Programmer's Guide*, published by COMPUTE! Books.

Custom Title Bars

```
10 FULLW 2 : CLEARW 2
20 title$="New Title" : 'Define title$ =
   program title.
30 GOSUB titlebar
40 END : 'Start your program here.
63000 titlebar : 'Custom title bar routine.
63010 a# = gb : gintin = PEEK(a#+8)
63020 POKE gintin+0,PEEK(systab+8) :
      POKE gintin+2,2
63030 s# = gintin+4 : title$ = title$ +
      CHR$(0)
63040 POKE s#,varptr(title$) : GEMSYS
      (105)
63050 RETURN
```

©

Looking Glass: Windows For The 64

James E. Hosek

This interesting program adds two new commands to Commodore 64 BASIC which let you create text windows and pull-down menus similar to those on Commodore 128, Atari ST, Amiga, and Macintosh computers. You can also add four text screens of information, including help screens.

"Looking Glass" is an all machine language utility that brings advanced windowing capabilities to the Commodore 64. Since it works as an extension to BASIC, you can use this program without understanding machine language at all.

To get started, type in the data from Program 1 using the "MLX" machine language entry program published elsewhere in this issue. Here are the starting and ending addresses you need for MLX:

Starting address: C000
Ending address: C62F

When you're finished entering all the data, be sure to use the MLX Save option to save at least one copy. If you want to try out the examples detailed below, be sure to save the data with the filename LG.

To use Looking Glass, load it with LOAD"LG",8,1 for disk or LOAD"LG",1,1 for tape. Activate it by typing SYS 49152 and pressing RETURN.

You now have two new BASIC statements for creating windows and menus. The OPENW (Open Window) statement opens a window on the text screen from any of five different sources (see below). The SAVEW (Save Window) statement saves the contents of an existing window into one of the four available workspaces which Looking Glass uses.

The new BASIC statements work either in immediate mode (when you're not running a program) or in program mode. Just like normal BASIC keywords, they can be abbreviated if you wish. The abbreviation for OPENW is O SHIFT-P W. The abbreviation for SAVEW is S SHIFT-A W.

To use Looking Glass in a program of your own, include these lines:

```
10 IF PEEK(49152)<>169 THEN LOAD
   "LG",8,1
20 SYS 49152
```

If you're using tape instead of disk, change the ,8,1 to ,1,1.

OPENW Opens Windows

Here is the general format for the OPENW statement:

OPENW *s,x,y,w,h,f*

OPENW can use from one to six parameters (values). The first parameter (*s* in this example) can range from 0 to 9 and must always be present. This value tells Looking Glass the *source* of the text which will appear in the new window. A source value of 0 designates the normal text screen (memory locations 1024–2023) as the source for the window. Opening a window from source 0 does not change what's currently displayed, since it merely copies the current contents of screen memory into the same locations.

Source values 1–4 designate one of the four workspaces which Looking Glass allocates underneath the 64's Kernal ROM. As you'll learn below, these areas initially contain garbage; the SAVEW command can be used to store meaningful information there.

Source values 5–9 have a special function. They automatically create a window the same size as the entire screen, using one of the four workspaces as a source of information. When you specify a source from 5 to 9, only the first parameter is relevant; Looking Glass ignores all additional parameters (see below).

Window Coordinates

The second and third parameters in an OPENW command (indicated by *x* and *y* in the previous example) locate the upper-left corner of the window you want to open. Specifying this corner's location effectively determines the screen position for the entire window. The horizontal *(x)* coordinate can range from 0 to 39, and the vertical *(y)* coordinate can range from 0 to 24.

The next two parameters (*w* and *h* in the previous example) represent the *width* and *height* of the window, respectively. The width value can range from 1 to 40, while the height value can range from 1 to 25. Note, however, that the maximum width and height for a given window depends on where its upper-left corner is located. For instance, if you locate the upper-left corner 10 columns from the left edge of the screen, you won't have room for a window that's 40 columns wide. To keep everything on the screen, you must make sure that a window's horizontal coordinate plus its width doesn't exceed 40, and that its vertical coordinate plus its height doesn't exceed 25.

The last parameter (*f* in the previous example) specifies the type of *frame* the window will have, and whether the window's contents will be normal or reverse video. A frame value of 0 creates a frameless window. A value of 2 selects a normal frame, and 4 creates a reversed frame. To make the window appear in reverse video, add 1 to any of the previous three values. The table below outlines the options for the frame parameter.

Table: Frame Parameter

0	No frame, normal window
1	No frame, reverse window
2	Normal frame, normal window
3	Normal frame, reverse window
4	Reverse frame, normal window
5	Reverse frame, reverse window

Any of the parameters for OPENW can be specified as a constant, variable, or arithmetic expression. For example, if S=1, then the statement OPENW S has the same effect as OPENW 1. If you omit a parameter, it defaults to the most recently used value (if any). To allow room for the frame, framed windows must have a width and height of at least three. Here are a few examples of legal OPENW commands:

```
OPENW 1,10,10,20,5
OPENW 4,,,25,10
OPENW 2,X,Y,10+X*2,5+Y*3,F
```

Saving With SAVEW

The SAVEW command saves the contents of a window in one of the special Looking Glass workspaces. This is useful when you need to save the contents of a window for further use and for certain other purposes which we'll explain below. Here is the general format for SAVEW, which takes only one parameter:

SAVEW *w*

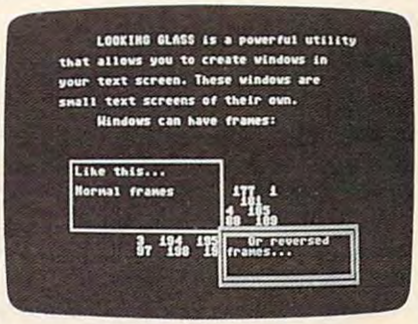
In this example *w* stands for *workspace*, and corresponds to the values used for the source in an OPENW command. Legal workspace values can range from 0 to 9. If you SAVEW with a value from 1 to 4, Looking Glass saves the contents of the current window in one of the four workspaces located under ROM. If you SAVEW with a value from 6 to 9, Looking Glass saves the entire display screen (which may be bigger than the current window) in the designated workspace.

Thus, after deciding which workspace to use, you have a basic choice between saving an entire screenful of information or saving only the contents of a window. Note that SAVEW stores the contents of a window or screen without disturbing what's already there. Values of 0 and 5 are legal for SAVEW, but have no visible effect since they simply store the contents of the current window or screen back into their present locations.

Working Inside Windows

After you open a window with a screen number of 0 to 4, certain restrictions apply. All text and output go only into the defined window area. Windows scroll separately from the rest of the screen, and a screen clear operation clears only the window. In immediate mode, commands can occupy only one physical line, without any wraparound at the window's edge. (If you wish to edit a program after creating a window, either press RUN/STOP–RESTORE or execute an OPENW command with a source value of 5 to 9.)

Windows also affect the behavior of the INST/DEL key and certain control codes for printing. When you type inside a window, either in direct mode or in response to an INPUT statement, the INST key (SHIFT–INST/DEL) always inserts a space at the cursor until the current line is full. DEL always deletes the character to the left of the cursor. If the cursor is at the beginning of a line, it wraps back to the end of the previous line, but does *not* pull any text with it. Looking Glass ignores CHR$(20) and CHR$(148) when they are printed to screens 0–4.



"Looking Glass" adds advanced windowing capabilities to Commodore 64 BASIC, making it easy to create and manipulate windows like this.

When you type inside a reversed window in immediate mode, control characters do not work when embedded in quotation marks. For example, typing PRINT "{HOME}" prints the letter *S* instead of homing the cursor as usual. To circumvent this problem, either type PRINT CHR$(147) or specify a nonreversed window. However, the control *keys* (RVS ON, CLR, BLK, and so on) work normally in every window.

When PRINTing inside a window, the SPC function works normally, since it refers to the current cursor position. The TAB function, however, refers to the left edge of the *screen*, not the left edge of the window, and may cause unexpected results unless the two edges coincide. Avoid using commas to separate items for printing (for instance, as in the statement PRINT X,Y,Z). When you separate printed items with commas, the computer arranges them into columns that are multiples of ten spaces—which may or may not fall inside the current window.

You will probably find the string functions (LEFT$, RIGHT$, MID$) and the semicolon (;) most useful for formatting text inside a window. If you exit a window by pressing RUN/STOP–RESTORE, don't forget to reactivate Looking Glass with SYS 49152 before trying to use OPENW or SAVEW again.

A Graphic Demonstration

Let's try some experiments to become familiar with windowing. First, activate Looking Glass as described above. Then clear the screen and enter the following statement in immediate mode (without a line number):

```
OPENW 1,5,5,30,15,2
```

A large boxful of random characters appears in the middle of the screen. Press SHIFT–CLR/HOME to get rid of the garbage characters. If you move around the window with the cursor keys, you'll notice that the window is actually only 28 × 15; the rest of the space is taken up by the frame. Enter a few direct commands to get a feel for how the window works. For instance, you may want to load a BASIC program, LIST it in the window, change the character colors, and so on.

Now type SAVEW 2 and press RETURN. This command stores the contents of the window in workspace 2. (Notice that you don't have to SAVEW a window to the same workspace that was used when you opened it.) Press SHIFT–CLR/HOME again, then enter OPENW 2. This retrieves the stored information from workspace 2. The frame color is the last color that you specified; all other window parameters default to their previous values.

To open a reversed window, enter this command:

```
OPENW 1,,,,,1: PRINT CHR$(147)
```

Note that the window is now a full 30 × 15. PRINTing CHR$(147) clears the window immediately so that no garbage appears. If you still have a program in memory, LIST it to confirm that the text indeed PRINTs in reverse video. To change the text color, press CTRL and any color key, then press SHIFT–CLR/HOME. The entire window changes to the selected color.

Press CTRL–RVS ON and type a few characters. Characters that are actually normal now appear in reverse mode. Next, enter OPENW 5 to leave the window and enter full screen mode. If you press SHIFT–CLR/HOME at this point, the whole screen is cleared. Enter OPENW 7. The previously stored text is now instantly recalled, along with the garbage that was not previously overwritten.

More Hints

The following line can be used to clear all four workspaces at the beginning of a program:

```
30 PRINT CHR$(147): SAVEW 6: SAVEW
   7: SAVEW 8: SAVEW 9
```

In some cases, you'll want two windows to overlap, but also be able to restore either window at any time. To accomplish this, save each window to a different workspace as soon as it is complete (that is, as soon as you're done printing in it). To restore the window, open it again with OPENW, using the same workspace number used when you saved it.

Sometimes it may be desirable to put a header or title in the frame of a window. The following example opens a 15 × 15 window with a normal frame and the header DIRECTORY:

```
100 OPENW 1,10,5,15,15,2: PRINT
    CHR$(147)
110 OPENW 0,10,5,15,1,0: PRINT
    "[RIGHT]DIRECTORY";
120 OPENW 0,11,6,13,13
```

Notice that line 120 opens from window 0 and that *x* and *y* are incremented by one, and *w* and *h* are decremented by two. In this case the *f* parameter defaults to zero, preventing Looking Glass from redrawing the frame and erasing the header.

While Looking Glass does not use any of the 64's BASIC programming space, it does use virtually all the RAM underneath the Kernal ROM, as well as RAM from locations 49152–50728 ($C000–$C628). The 64's BASIC ROM is also copied to underlying RAM and modified.

The more you learn about how Looking Glass works, the more uses you'll find for it. A pull-down menu, for instance, is simply a window located on the top edge of the screen. Program 2 demonstrates how to create nondestructive pull-down menus as well as many other unique effects. Once you master the techniques involved, you'll probably think of even more applications.

Program 1: Looking Glass

For instructions on entering this listing, please refer to the "MLX" article published in this issue of COMPUTE!.

```
C000:A9 51 8D 08 03 A9 C3 8D B0
C008:26 03 A9 E6 8D 02 03 A9 25
C010:DD 8D 04 AC A9 F6 8D 18 8B
C018:03 A9 C5 8D 19 03 A9 C0 FF
C020:8D 09 03 A9 C3 8D 27 03 4B
C028:A9 C4 8D 03 03 A9 C5 8D 69
C030:05 AC A9 00 85 FB A9 A0 A4
C038:85 FC A0 00 B1 FB 91 FB 6C
C040:E6 FB D0 F8 E6 FC A5 FC 51
C048:C9 C0 D0 F0 A9 76 85 01 3B
C050:60 20 73 00 8D 16 C6 C9 94
C058:9F D0 0E A0 01 B1 7A C9 37
C060:57 D0 18 A9 06 85 FB D0 6E
C068:18 C9 94 D0 0E A0 01 B1 AE
C070:7A C9 57 D0 06 A9 01 85 F7
C078:FB D0 06 20 79 00 4C E7 3B
C080:A7 20 73 00 20 73 00 88 A3
C088:98 48 20 9E AD A9 00 2A 94
C090:20 90 AD 20 1B BC 20 BF C9
C098:B1 68 A8 A5 65 99 01 C6 D6
C0A0:20 79 00 C9 2C D0 1C 20 2A
C0A8:73 00 C8 C9 2C D0 02 F0 33
C0B0:F6 C4 FB F0 0E D0 D1 AC 71
C0B8:09 C6 20 CA F1 88 C0 02 56
C0C0:D0 F8 60 AD 01 C6 30 0E 61
C0C8:C9 0A 10 0A AE 16 C6 E0 90
C0D0:94 D0 0D 4C 2A C3 A9 00 EA
C0D8:8D 19 C6 A2 0E 6C 00 03 8F
C0E0:AD 01 C6 C9 05 30 07 18 FD
C0E8:6E 19 C6 4C DE C2 38 6E 66
C0F0:19 C6 A0 04 B9 01 C6 99 FD
C0F8:06 C6 88 D0 F7 AD 07 C6 98
C100:18 6D 09 C6 C9 29 10 CE 5A
C108:AD 08 C6 18 6D 0A C6 C9 A9
C110:1A 10 C3 A9 00 8D 17 C6 E2
C118:AD 06 C6 4A 8D 18 C6 6E 3A
C120:17 C6 C9 03 10 B0 C9 00 21
C128:F0 02 A9 02 CD 09 C6 10 2A
C130:A5 CD 0A C6 10 A0 AD 18 1E
C138:C6 C9 00 D0 03 4C F6 C1 97
C140:AD 18 C6 C9 01 F0 04 A9 93
C148:12 D0 02 A9 92 20 CA F1 80
C150:AE 08 C6 AC 07 C6 18 86 DA
C158:02 20 F0 FF E6 02 A9 B0 46
C160:20 CA F1 A9 60 20 B7 C0 33
C168:A9 AE 20 CA F1 AD 0A C6 3E
C170:38 E9 02 85 FB A6 02 AC 4E
C178:07 C6 18 20 F0 FF A9 7D 8E
C180:20 CA F1 A5 D3 18 6D 09 42
C188:C6 A8 88 88 84 D3 A9 7D 77
C190:20 CA F1 E6 02 C6 FB D0 77
C198:DC A6 02 AC 07 C6 18 20 E2
C1A0:F0 FF A9 AD 20 CA F1 A9 66
C1A8:60 20 B7 C0 AD 08 C6 18 9A
C1B0:6D 0A C6 C9 19 D0 23 AD E2
C1B8:07 C6 18 6D 09 C6 C9 28 6A
C1C0:D0 18 AD 18 C6 C9 02 F0 3C
C1C8:04 A9 7D D0 02 A9 FD 8D B5
C1D0:E7 07 AD 86 02 8D E7 DB 1A
C1D8:D0 05 A9 BD 20 CA F1 EE 16
C1E0:07 C6 EE 08 C6 CE 09 C6 42
C1E8:CE 09 C6 CE 0A C6 CE 0A EE
C1F0:C6 A9 00 8D 18 C6 A9 00 4A
C1F8:8D 21 C6 AC 01 C6 A9 04 A9
C200:8D 1F C6 B9 0C C6 8D 22 41
C208:C6 20 61 C2 AC 01 C6 A9 F1
C210:D8 8D 1F C6 B9 11 C6 8D E2
C218:22 C6 20 61 C2 A9 92 4D A9
C220:17 C6 20 CA F1 AE 08 C6 B4
C228:AC 07 C6 18 20 F0 FF AD 92
C230:07 C6 18 6D 09 C6 A8 8C 06
C238:1B C6 88 8C 1A C6 AD 08 26
```

```
C240:C6 18 6D ØA C6 A8 8C 1D 8C
C248:C6 88 8C 1C C6 AC Ø8 C6 66
C250:B9 D9 ØØ Ø9 8Ø 99 D9 ØØ D7
C258:C8 CC 1D C6 3Ø F2 4C DB 47
C260:C2 AC ØA C6 A9 ØØ 85 FC 75
C268:AD Ø8 C6 ØA ØA 18 6D Ø8 D3
C270:C6 ØA ØA 26 FC ØA 26 FC D8
C278:18 6D Ø7 C6 85 FB 9Ø Ø2 F1
C280:E6 FC 18 A5 FB 6D 21 C6 B4
C288:85 FD AD 22 C6 65 FC 85 73
C290:FE 18 AD 1F C6 65 FC 85 8E
C298:FC 78 A5 Ø1 48 A9 75 85 D8
C2A0:Ø1 98 AA AC Ø9 C6 88 B1 13
C2A8:FD 2C 1E C6 1Ø Ø6 AD 23 7F
C2B0:C6 ØD 17 C6 91 FB 88 1Ø C9
C2B8:EE 18 A9 28 65 FB 85 FB 95
C2C0:A9 ØØ 65 FC 85 FC 18 A9 91
C2C8:28 65 FD 85 FD A9 ØØ 65 CF
C2D0:FE 85 FE CA DØ CD 68 85 D7
C2D8:Ø1 58 6Ø 4C 7B CØ 38 E9 FE
C2E0:Ø5 8D Ø1 C6 A9 ØØ 8D 19 5A
C2E8:C6 2Ø EF C2 4C F6 C1 A9 6F
C2F0:ØØ 8D Ø7 C6 8D Ø8 C6 A9 EA
C2F8:28 8D Ø9 C6 A9 19 8D ØA 5A
C300:C6 6Ø A9 ØØ 8D 21 C6 AC 63
C308:Ø1 C6 A9 Ø4 8D 22 C6 B9 73
C310:ØC C6 8D 1F C6 2Ø 61 C2 2F
C318:AC Ø1 C6 A9 D8 8D 22 C6 B1
C320:B9 11 C6 8D 1F C6 2Ø 61 3Ø
C328:C2 6Ø C9 Ø5 1Ø Ø6 2Ø Ø2 8D
C330:C3 4C 7B CØ 38 E9 Ø5 8D 29
C338:Ø1 C6 A2 Ø3 BD Ø7 C6 48 56
C340:CA 1Ø F9 2Ø EF C2 2Ø Ø2 3F
C348:C3 E8 68 9D Ø7 C6 E8 EØ D8
C350:Ø5 DØ F7 4C 7B CØ 8E 24 72
C358:C6 8C 25 C6 AC ØA C6 CØ 53
C360:Ø1 FØ 25 A9 D8 8D 1F C6 E5
C368:8D 22 C6 A9 28 8D 21 C6 33
C370:88 2Ø 64 C2 A9 Ø4 8D 1F 94
C378:C6 8D 22 C6 A9 28 8D 21 A1
C380:C6 AC ØA C6 88 2Ø 64 C2 94
C388:38 6E 1E C6 A9 2Ø 8D 23 Ø4
C390:C6 A9 ØØ 85 FC AD 1C C6 DB
C398:ØA ØA 6D 1C C6 ØA 26 FC BE
C3A0:ØA 26 FC ØA 26 FC 18 6D B9
C3A8:Ø7 C6 85 FB A9 ØØ 69 Ø4 F9
C3B0:65 FC 85 FC AØ Ø1 2Ø 99 8D
C3B8:C2 ØE 1E C6 AE 24 C6 AC 95
C3C0:25 C6 6Ø Ø8 2C 19 C6 3Ø 9C
C3C8:Ø4 28 4C CA F1 85 Ø2 8E CA
C3D0:26 C6 8C 27 C6 38 2Ø FØ 69
C3D8:FF A5 Ø2 C9 ØD DØ 2Ø A9 3C
C3E0:ØØ 85 D4 2C 17 C6 1Ø Ø2 1D
C3E8:A9 12 85 C7 EC 1C C6 3Ø 8C
C3F0:Ø8 2Ø 56 C3 AE 1C C6 1Ø ØF
C3F8:Ø1 E8 AC Ø7 C6 1Ø 4Ø C9 Ø2
C400:8D FØ DC 48 A5 D4 FØ Ø4 13
C408:68 4C B8 C4 68 C9 93 DØ 9E
C410:24 38 6E 1E C6 A9 2Ø 8D 14
C418:23 C6 A9 Ø4 8D 1F C6 2Ø FØ
C420:61 C2 AD 86 Ø2 8D 23 C6 7C
C428:A9 D8 8D 1F C6 2Ø 61 C2 9C
C430:4E 1E C6 1Ø Ø4 C9 13 DØ 8Ø
C438:16 AC Ø7 C6 AE Ø8 C6 18 8Ø
C440:2Ø FØ FF AE 26 C6 AC 27 CD
C448:C6 A5 Ø2 28 58 18 6Ø C9 ØF
C450:91 DØ Ø7 EC Ø8 C6 FØ EB AF
C458:DØ 19 C9 11 DØ ØC EC 1C 87
C460:C6 1Ø Ø2 3Ø ØE 2Ø 56 C3 F5
C468:BØ D9 C9 1D DØ 1F CC 1A 82
C470:C6 1Ø ØA 28 AE 26 C6 AC 6D
C478:27 C6 4C CA F1 EC 1C C6 BF
C480:3Ø Ø5 2Ø 56 C3 BØ Ø1 E8 98
C488:AC Ø7 C6 1Ø B2 C9 9D DØ CC
C490:12 CC Ø7 C6 FØ Ø2 DØ DB BØ
C498:EC Ø8 C6 FØ A6 AC 1A C6 65
C4A0:CA 1Ø 9C C9 14 FØ 9C C9 2B
C4A8:94 FØ 98 C9 12 DØ Ø5 4D 93
C4B0:17 C6 DØ BF C9 92 FØ F7 FF
C4B8:CC 1A C6 FØ Ø2 DØ B4 EC CØ
C4C0:1C C6 FØ 1Ø 2Ø CA F1 E8 22
C4C8:B5 D9 Ø9 8Ø 95 D9 AC Ø7 41
C4D0:C6 4C 3F C4 EC Ø8 C6 FØ ØB
C4D8:EF 2Ø 56 C3 CA 18 2Ø FØ 51
C4E0:FF A5 Ø2 4C C4 C4 2C 19 83
C4E8:C6 3Ø Ø3 4C 83 A4 2Ø F4 EA
C4F0:C4 4C 86 A4 A4 D3 B1 D1 B4
C4F8:85 FE A9 ØØ 85 CC 2Ø E4 BE
C500:FF AA FØ F6 48 A9 Ø1 85 34
C508:CC A5 FE A4 D3 91 D1 AD C3
C510:87 Ø2 91 F3 68 C9 ØD FØ C6
C518:11 C9 8D FØ ØD C9 14 FØ Ø8
C520:6F C9 94 FØ 37 2Ø D2 FF 57
C528:9Ø CA AC Ø7 C6 AE Ø9 C6 7E
C530:A9 ØØ 85 FB A9 Ø2 85 FC 5E
C538:88 E8 B1 D1 8C 27 C6 4D 71
C540:17 C6 A8 3Ø 19 C9 2Ø 1Ø 61
C548:Ø5 18 69 4Ø 9Ø 16 C9 4Ø 3E
C550:1Ø Ø2 3Ø 1Ø C9 6Ø 1Ø F1 4D
C558:Ø9 8Ø DØ Ø8 FØ 5D 29 7F F1
C560:C9 4Ø 1Ø E5 AØ ØØ 91 FB 65
C568:AC 27 C6 E6 FB C8 CA DØ C4
C570:C9 AØ ØØ 98 91 FB C6 FB 98
C578:B1 FB 29 3F C9 2Ø DØ Ø6 6B
C580:A9 ØØ 91 FB FØ FØ A9 ØD 7E
C588:2Ø D2 FF A2 ØØ AØ Ø2 6Ø E9
C590:A4 D3 CC Ø7 C6 FØ 14 B1 41
C598:D1 88 91 D1 C8 C8 CC 1B 9C
C5A0:C6 DØ F4 88 A9 2Ø ØD 17 E9
C5A8:C6 91 D1 A6 D4 A9 ØØ 85 73
C5B0:D4 A9 9D 2Ø D2 FF 86 D4 3F
C5B8:4C F4 C4 AC 1A C6 B1 D1 2C
C5C0:29 7F C9 2Ø DØ 12 88 B1 8D
C5C8:D1 C8 91 D1 88 C4 D3 DØ 8E
C5D0:F5 A9 2Ø 2Ø D2 FF 9Ø D3 53
C5D8:A9 94 4C 25 C5 2C 19 C6 12
C5E0:3Ø Ø3 4C 6Ø A5 38 2Ø FØ 14
C5E8:FF 98 18 ED Ø7 C6 85 FA D5
C5F0:2Ø F4 C4 A6 FA 6Ø 24 91 FF
C5F8:3Ø Ø4 18 6E 19 C6 4C 47 4B
C600:FE ØØ ØØ ØØ 28 19 ØØ ØØ B2
C608:ØØ ØØ ØØ ØØ Ø4 EØ E8 FØ FB
C610:F8 D8 E4 EC F4 FC ØØ ØØ 57
C618:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ A5
C620:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ AD
C628:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ B5
```

Program 2: Window Demonstration

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" published in this issue of COMPUTE!.

```
MC 1Ø IFPEEK(49152)<>169THENLO
      AD"LG",8,1
SQ 2Ø SYS49152
JK 3Ø POKE53280,11:POKE53281,1
      1:PRINT"{8}{CLR}";CHR$(1
      4)
QS 4Ø OPENW5:SAVEW6:SAVEW7:SAV
      EW8:SAVEW9
XP 5Ø PRINT"{CLR}{OFF}
      {5 SPACES}LOOKING
      {SHIFT-SPACE}GLASS IS A
      {SPACE}POWERFUL UTILITY"
FP 6Ø PRINT"THAT ALLOWS YOU TO
       CREATE WINDOWS IN":PRIN
      T
HB 7Ø PRINT"YOUR TEXT SCREEN.
      {SPACE}THESE WINDOWS ARE
      ":PRINT
JC 8Ø PRINT"SMALL TEXT SCREENS
       OF THEIR OWN.":GOSUB74Ø
EX 9Ø PRINT"{WHT}";:OPENW1,1Ø,
      15,19,7,Ø:PRINT"{CLR}";:
      FORX=1TO2ØØ:PRINTX;:NEXT
SB 1ØØ SAVEW1: OPENW5:PRINT"
       {HOME}{CYN}{8 DOWN}
       {5 SPACES}WINDOWS CAN H
       AVE FRAMES:":GOSUB74Ø
SC 11Ø PRINT"{6}";:OPENW2,1,12
       ,21,8,2:PRINT"{CLR}LIKE
        THIS..."
DK 12Ø PRINT"{DOWN}NORMAL FRAM
       ES{YEL}":SAVEW2:OPENW2,
       21,19,19,6,4
PP 13Ø PRINT"{CLR}{3 SPACES}OR
        REVERSED{3 SPACES}FRAM
       ES...":SAVEW2:GOSUB74Ø:
       OPENW5
FH 14Ø PRINT"{WHT}{HOME}
       {8 DOWN}{5 SPACES}PLUS
       {SPACE}WINDOWS CAN BE R
       EVERSED:{CYN}":GOSUB74Ø
MQ 15Ø OPENW3,5,11,15,14,5:PRI
       NT"{CLR}LIKE THIS...":G
       OSUB74Ø
PD 16Ø FORX=1TO1ØØ:PRINTX;:NEX
       T:PRINT:SAVEW3:GOSUB74Ø
       :OPENW4,Ø,Ø,4Ø,1Ø,Ø:PRI
       NT"{CLR}"
SF 17Ø PRINT"{7}{OFF}
       {5 SPACES}EVEN THOUGH T
       HE WINDOWS OVERLAP,":PR
       INT
KD 18Ø PRINT"THEY ARE NON DEST
       RUCTIVE:":GOSUB74Ø
BD 19Ø OPENW1,1Ø,15,19,7,Ø:GOS
       UB74Ø:PRINT"{6}"
CR 2ØØ OPENW2,1,12,21,8,2:GOSU
       B74Ø:PRINT"{YEL}"
PG 21Ø OPENW2,21,19,19,6,4:GOS
       UB74Ø:PRINT"{CYN}";:OPE
       NW3,5,11,15,14,5:GOSUB7
       4Ø
QH 22Ø OPENW5:SAVEW5:OPENW1,Ø,
       Ø,4Ø,25,5:PRINT"{CLR}"
EQ 23Ø PRINT"{DOWN}{5 RIGHT}ON
       E OF THE POSSIBLE USES
       {SPACE}OF{DOWN}"
EP 24Ø PRINT" LOOKING GLASS IS
        TO CREATE PULL DOWN
       {DOWN}"
HX 25Ø PRINT" MENUS:"
FR 26Ø PRINT"{DOWN}{3 SPACES}
       {DOWN}TYPE P, T, OR F T
       O PULL DOWN MENU"
BM 27Ø PRINT"{DOWN}{3 SPACES}P
       RESS {RVS}RETURN{OFF} T
       O GO ON:":FR=Ø:PR=Ø:TV=
       Ø
EG 28Ø OPENWØ,Ø,Ø,4Ø,1,1:PRINT
       "{2 RIGHT}PRESIDENTS
       {4 RIGHT}TV SHOWS
       {6 RIGHT}FRUITS";
CC 29Ø OPENW5:SAVEW7
XC 3ØØ X=PEEK(2Ø3):IFX=64THEN3
       ØØ
GS 31Ø IFX=1THEN56Ø
BM 32Ø IFX=41THEN36Ø
PE 33Ø IFX=22THEN43Ø
FK 34Ø IFX=21THEN48Ø
FH 35Ø GOTO3ØØ
HF 36Ø PRINT"{GRN}":OPENW4,1,Ø
       ,12,17,5:IFPRTHEN41Ø
PK 37Ø PRINT"{CLR}PRESIDENTS"
BQ 38Ø PRINT"WASHINGTONLINCOLN
       ":PRINT"ROOSEVELT":PRIN
       T"NIXON":PRINT"JOHNSON"
SH 39Ø PRINT"JEFFERSON":PRINT"
       FORD":PRINT"CARTER":PRI
       NT"REAGAN"
GX 4ØØ PRINT"ADAMS":PRINT"MADI
       SON":PRINT"GRANT"
JS 41Ø X=PEEK(2Ø3):IFX=41THEN4
       1Ø
BK 42Ø PR=-1:SAVEW4:OPENW7:GOT
       O3ØØ
GK 43Ø PRINT"{YEL}":OPENW4,14,
       Ø,12,12,5:IFTVTHEN46Ø
BA 44Ø PRINT"{CLR} TV
       {SHIFT-SPACE}SHOWS
       {DOWN}":PRINT"A TEAM":P
       RINT"COSBY SHOW";
GM 45Ø PRINT"NIGHT{DOWN}
```

```
        {3 LEFT}COURT":PRINT"FA
        MILY{DOWN}{4 LEFT}TIES"
        :PRINT"LATE NIGHT";
XF 460 X=PEEK(203):IFX=22THEN4
        60
GD 470 TV=-1:SAVEW4:OPENW7:GOT
        O300
KJ 480 PRINT"{7}":OPENW4,27,0,
        12,23,5:IFFRTHEN540
PH 490 PRINT"{CLR}{2 SPACES}FR
        UITS"
JD 500 PRINT"{DOWN}APPLES":PRI
        NT"ORANGES":PRINT"BANAN
        AS":PRINT"PEARS":PRINT"
        LEMONS"
HQ 510 PRINT"KUMQUATS":PRINT"K
        IWI FRUITWATERMELONGRAP
        EFRUITTANGERINE STRAWBE
        RRY";
CQ 520 PRINT"PLUM":PRINT"PEACH
        ":PRINT"BLUEBERRY":PRIN
        T"APRICOT":PRINT"RASPBE
        RRY"
KF 530 PRINT"PINEAPPLE CHERRY"
        ;
DD 540 X=PEEK(203):IFX=21THEN5
        40
HD 550 FR=-1:SAVEW4:OPENW7:GOT
        O300
KX 560 PRINT"{CLR}{10 DOWN}
        {3 RIGHT}{RVS}PLEASE PU
        T ON YOUR SAFETY GOGGLE
        S"
DG 570 PRINT"{DOWN}{3 RIGHT}
        {7 SPACES}THEN PRESS RE
        TURN:":POKE198,0
QG 580 GETA$:IFA$<>CHR$(13)THE
        N580
ES 590 POKE198,0:PRINT"{CLR}":
        C$(0)="{CYN}":C$(1)="
        {YEL}":C$(2)="{GRN}":C$
        (3)="{WHT}"
SE 600 FORX=0TO70:PRINT"LOOKIN
        G GLASS "C$(XAND3);:NEX
        T:SAVEW6
DR 610 PRINT"{CLR}{RVS}":FORX=
        1TO23:PRINT"{39 +}":NEX
        T
JG 620 OPENW0,0,0,40,25,4
PP 630 OPENW0,6,5,29,15,0:PRIN
        T"{WHT}{CLR}":FORX=1TO1
        3:PRINT"{28 *}"
QX 640 NEXT:OPENW0,5,5,30,15,4
        :PRINT"{YEL}";:SAVEW7
GA 650 FORX=1TO10:OPENW1,19-X,
        12-X,2*X+2,2*X+2,4:GOSU
        B750:NEXT
XM 660 FORX=0TO9:OPENW2,9+X,2,
        1,22,0:OPENW1,10+X,2,21
        -X,22,4:GOSUB750:NEXT
MG 670 FORX=0TO9:OPENW2,18,2+X
        ,13,1,0:OPENW1,19,3+X,1
        2,21-X,4:GOSUB750:NEXT
PD 680 FORX=1TO10:OPENW2,20-X,
        24-X,12,1,0:OPENW2,31-X
        ,13-X,1,11,0
JM 690 OPENW1,19-X,12-X,12,12,
        4:GOSUB750:NEXT:SAVEW7
FM 700 FORX=1TO11:OPENW7:GOSUB
        750:GOSUB750:OPENW6:GOS
        UB750:GOSUB750:NEXT
BB 710 FORX=1TO50:PRINT"{DOWN}
        ";:NEXT
QJ 720 PRINT"PLEASE REMOVE YOU
        R SAFETY GOGGLES NOW.
        {10 DOWN}"
BD 730 END
QC 740 FORX=1TO1500:NEXT:RETUR
        N
BR 750 FORZ=1TO100:NEXT:RETURN
```
©

ST Hints & Tips

George Miller, Assistant Technical Editor

Here are some interesting tricks for setting up autobooting programs, customizing your GEM desktop, reading a joystick from ST BASIC, and souping up BASIC's performance with machine language subroutines. All the techniques work on the 520ST and 1040ST.

The Atari ST series computers are extremely powerful and complex machines. The numerous demo programs which are widely available offer only small peeks at the true capabilities of these computers. For programmers, however, the ST's power can be frustrating because it's so elusive. Virtually no technical documentation is supplied with the ST, and the two languages it comes with—Logo and ST BASIC—have their shortcomings.

If you invest $300 for an Atari development system package, you receive an assembler, a C compiler, and a huge mass of documentation on the Graphics Environment Manager (GEM), but most of it is not even ST-specific—it refers to GEM as implemented on the IBM PC.

However, careful study of this mountain of paper can reveal quite a few "secrets" about the ST. We'll let you in on a few of these tricks which enhance the power of your computer.

Autobooting Programs

How you ever wished that a certain program—perhaps a RAM disk utility, or an application, or a language—could run automatically when you start up your ST? This feature would be especially handy if you need to set up a disk for someone who wants to run a program without understanding anything more than how to turn on the computer.

The eight-bit Atari computers can automatically load and run programs by using AUTORUN.SYS files. Apple has the HELLO program, PC-DOS and AmigaDOS have batch files, and the Commodore 128 has provision for autobooting. Although it's not documented, so does your ST. Clues on how to create an autoexecute file can be found in GEMDOS.

As part of the initialization sequence, the ST looks for a folder called AUTO on the boot disk. Any files with a .PRG extender found in the AUTO folder are executed in sequence. These files are known as COMMAND.PRG files.

It's very easy to set up an autoboot program. Place your boot disk in your drive, then point to the File heading on the menu bar. Select the New Folder option and create a folder named AUTO.

Move any program you want to autoboot into this folder. Any time you boot your ST from this disk, the program you placed in the AUTO folder will automatically run. This technique works with TOS in ROM or with the earlier disk-loaded TOS. There may be a problem, however, with autobooting some programs when using the high-resolution monochrome mode. Otherwise, it's the most foolproof autorun system yet.

Customizing The Desktop

Have you ever tried renaming your disk icons using the Install Drive option from the Options menu? Some characters can't be used. For instance, it isn't possible to name an icon Disk A because lowercase letters and spaces are not permitted. Also, you can't do anything with the trash can.

However, there is a way to change the names to anything you want. After saving your desktop, you can edit the file which stores the information for these options—DESKTOP.INF. For now we'll only change the icon names using this technique. Be careful to not change any other characters in the file.

First, you'll need a text editor such as *Mince* or *EMACS*, or even a word processor, like *ST Writer*. If you're using a word processor, set the left and top margins to zero.

The job itself is rather easy. Load the file DESKTOP.INF. It should look something like Figure 1.

Figure 1: DESKTOP.INF

```
#a000000
#b001100
#c77700070007000700552005055522207
 70557075057705504112306
#d
#E 9B 03
#W 00 00 0C 01 1D 16 08 A:\*.*@
#W 00 00 28 01 1F 17 00 @
#W 00 00 0E 09 2A 0B 00 @
#W 00 00 0F 0A 2A 0B 00 @
#M 00 02 00 FF A FLOPPY DISK@ @
#M 00 03 00 FF B FLOPPY DISK@ @
#T 00 07 02 FF TRASH CAN@ @
#F FF 04 @ *.*@
#D FF 01 @ *.*@
#G 03 FF *.PRG@ @
#F 03 04 *.TOS@ @
#P 03 04 *.TTP@ @
```

Each character in this file is information about your desktop. Any change will affect what you see on the desktop and even how your ST functions to a certain extent. Use caution, since some changes might not yield the results you expect. To be safe, make sure you're working with a backup copy of your boot disk. Store the original in a safe location. This is always a good idea when experimenting with any file on a disk, and especially when modifying files that control the operation of your ST.

Now, move the cursor to the first line which begins with #M. Change the text, replacing the words FLOPPY DISK, so the line reads like this:

```
#M 00 02 00 FF A Disk A@ @
```

Then change the next line to:

```
#M 00 02 00 FF B Disk B@ @
```

If you want, you may change the name of the trash can icon. I called mine *Black Hole!* as a constant reminder that unlike the Amiga or Macintosh, the ST trash can does not let you easily recover files which are deleted. (There are some disk utilities available which allow you to recover trashed files, under limited conditions.)

To change the trash can icon, modify the next line to read:

```
#T 00 07 02 FF Black Hole!@ @
```

The revised DESKTOP.INF file should be similar to Figure 2.

Figure 2: Revised DESKTOP.INF

```
#a000000
#b001100
#c77700070007000700552005055522207
 70557075057705504112306
#d
#E 9B 03
#W 00 00 0C 01 1D 16 08 A:\*.*@
#W 00 00 28 01 1F 17 00 @
#W 00 00 0E 09 2A 0B 00 @
#W 00 00 0F 0A 2A 0B 00 @
#M 00 02 00 FF A Disk A@ @
#M 00 03 00 FF B Disk B@ @
#T 00 07 02 FF Black Hole!@ @
#F FF 04 @ *.*@
#D FF 01 @ *.*@
#G 03 FF *.PRG@ @
#F 03 04 *.TOS@ @
#P 03 04 *.TTP@ @
```

Finally, save the file back to the disk as DESKTOP.INF. The file must be saved in ASCII format, so make sure your text editor or word processor has this feature. If you're using *ST Writer* or some other word processors, it may be necessary to print the file to the disk in order to save it in ASCII format.

Reading The ST Joystick

ST BASIC is a fairly generic BASIC that has very few ST-specific commands. One of the most noticeably missing commands when you're trying to write a game is a function for reading the joystick. The ST works with any of the joysticks sold for the eight-bit Atari and Commodore computers, but there's no STICK or STRIG functions as in eight-bit Atari BASIC.

Actually, a joystick command does exist in the ST, but it's hidden deep within GEMDOS in the BIOS (Basic Input/Output System). This is an area not readily available from ST BASIC without using a few special techniques.

One easy way to find out what the joystick is doing is to ask the Intelligent Keyboard Device (IKBD). The keyboard has its own microprocessor, a 6301 chip, which is a member of the 6800 family. The keyboard processor is really a small computer system, with input/output lines, RAM, ROM, and even a serial interface which handles traffic to and from the 68000 central processing unit. The 68000 is not responsible for polling the keyboard continuously for activity. The 6301 notifies the 68000 via an interrupt when anything needs processing. In addition to reading the keyboard, the 6301 also reads the mouse, the joystick, and performs other functions.

The ST's link to the keyboard processor is through a chip called an ACIA (Asynchronous Communications Interface Adapter). The control register for the keyboard ACIA is located at memory address $FFFC00 in the ST, and the data register is at location $FFFC02. If you've moved to the ST from an earlier eight-bit computer, those may be the biggest hexadecimal numbers you've ever seen. Remember that the 68000 microprocessor in the ST has 24 address lines, enough for over 16,000,000 bytes of memory, as compared to the 65,536-byte maximum for eariler computers with only 16 address lines. For the ST you must become accustomed to seeing hexadecimal addresses that are six digits long.

The following program is a short ST BASIC routine to read the values of the joystick plugged into port 1 (the rear joystick connector).

```
70 POKE &hfffc02,&h0012 'turn off
   mouse
80 POKE &hfffc02,&h0014 : joystick =
   PEEK(&hfffc02)
90 IF joystick = 511 THEN ? "north"
100 IF joystick = 2559 THEN ?
    "northeast"
110 IF joystick = 2303 THEN ? "east"
120 IF joystick = 2815 THEN ?
    "southeast"
130 IF joystick = 767 THEN ? "south"
140 IF joystick = 1791 THEN ?
    "southwest"
150 IF joystick = 1279 THEN ? "west"
160 IF joystick = 1535 THEN ?
    "northwest"
170 IF joystick < 0 THEN ? "fire button"
180 POKE &hfffc02,&h0008 'turn on
    mouse
190 GOTO 70
```

Line 70 sends a command to the IKBD, via the data register at $FFFC02, instructing it to turn off the mouse. (Note that ST BASIC uses &h to indicate hexadecimal numbers.)

Line 80 sends a command via the same address to turn on the joystick. Every movement of the joystick is reported to the processor. The joystick position is read by PEEKing the value returned in $FFFC02.

Lines 90–170 interpret the values returned from the IKBD.

Line 180 turns the mouse back on again. This should be done before exiting the program so the user will have control of the mouse when returning to BASIC or the desktop.

Line 190 makes the routine an infinite loop, so you'll need to press CTRL-C to stop this demonstration. If the mouse pointer isn't visible on the screen when the program stops, enter the following line and press RETURN to make the pointer reappear:

```
POKE &hfffc02,&h0008
```

To adapt this routine for use in your own programs, replace line 190 with 190 RETURN, then use GOSUB 70 to call the routine. Replace the PRINT statements in lines 90–170 with statements to perform the desired actions when the joystick is pressed in the indicated direction.

Mixing BASIC And Machine Language

To add real speed and power to any BASIC, it's often necessary to use machine language routines for certain tasks. In ST BASIC, machine language routines can be run using the CALL statement. The syntax for CALL is:

CALL *address variable,parameter list*

The *address variable* is a variable which holds the memory address of the beginning of the machine language routine. This location may be the address where the routine was loaded using the BLOAD command, or the address where the ML routine was POKEd. The *parameter list* is a list of values which can be passed to the ML routine. Some routines don't require any values to be passed, so this is optional.

The program below demonstrates how to POKE an ML routine into a variable, then use the VARPTR function to find the address to CALL.

As your library of ML routines expands, you'll find this method useful. Although the example program does nothing but print the letter A on the left side of the menu bar, it does demonstrate that ML routines give you full access to the ST, since the menu bar is usually off-limits to BASIC.

```
110 CLEARW 2 : FULLW 2
120 GOSUB init
130 ' ML opcodes in DATA statements
140 DATA &h3f3c,&h0041,&h3f3c,
    &h0002,&h4e41,&h588f
150 DATA &h3f3c,&h000d,&h3f3c,
    &h0002,&h4e41,&h588f
160 DATA &h3f3c,&h000a,&h3f3c,
    &h0002,&h4e41,&h588f,&h4e75
170 FOR i = 1 TO 19 : READ a : POKE
    x+(i*2),a : NEXT : 'POKE ml into
    ml$
180 CALL x
190 END
200 init : ml$="This is a dummy
    variable."
210 x = VARPTR (ml$)
220 RETURN
```

These tricks demonstrate only a small part of the ST's potential. Carefully studying the documentation reveals that some extremely powerful programming techniques are lurking just below the surface. If you're a curious programmer, explore GEM for ways to use the ST's features from within the tight BASIC framework. ©

Minding Memory From BASIC

D. W. Neuendorf

Are your programs fighting wars with each other for control of memory? Would you like to find a safe, protected place in RAM for machine language subroutines and other data in your BASIC programs? Here's how to use the memory management functions of PC-DOS to avoid conflicts and maximize the amount of memory available to BASIC. For the IBM PC, PCjr, and compatibles with DOS 2.0 or higher.

Over the past year, memory management in PC-DOS has become an important issue. The new desktop tools and coresident programs are designed to wait in the background to be called during the operation of another program. A number of these utilities may be lurking in memory at once, and programmers can't predict which other programs will be present with their own. The result can be memory conflicts and system crashes.

The designers of PC-DOS anticipated this situation to some extent. DOS 2.0 and later versions contain several function calls designed to give the operating system control over how the computer's memory is divided among programs residing in memory simultaneously. The most basic of these functions simply attempt to allocate and deallocate blocks of memory at a program's request. These DOS calls are readily available to machine language programmers, just like all other machine-level resources.

BASIC programmers, on the other hand, have no direct access to many DOS functions. But as we'll see, there are ways for BASIC programs to call on DOS to perform these memory management tasks.

Translating ML To BASIC

There are two DOS functions we're interested in—one for allocating memory and another for deallocating memory.

In machine language, both functions are called by placing a *function number* in the microprocessor's AH register and calling interrupt 21h. (Function numbers indicate to DOS which function is being called. The interrupt then performs the function.) The numbers are 48h for the allocate function and 49h for the deallocate function.

In addition to these numbers, each function call requires that you pass an argument. The allocate function requires the number of 16-byte *paragraphs* of memory to be allocated. This number must be placed in the microprocessor's BX register. The deallocate function requires the segment address of a block to be deallocated. This number must be placed in the ES register.

After each function is performed, it returns a value. The allocate function returns, via the AX register, either the segment address of an allocated block or an error code (7 or 8 plus a set carry bit) if the function was unsuccessful. The deallocation routine returns nothing if successful, but sets the carry bit and returns an error code (7 or 9) if unsuccessful. For those who are interested, Programs 1 and 2 show the assembler code necessary to call these functions.

Program 3 shows how to call these functions from BASIC. Since the allocate routine is not available initially and therefore can't allocate space for itself, the program reserves a few bytes for it just above BASIC (using the CLEAR statement in line 10). Once the allocate routine has been installed (lines 40–60), it can be used to get memory from DOS for machine language routines and other data. An example of its use is the call in line 70, which gets the segment address of a memory block for the deallocate routine.

Finally, line 120 shows an example of using the deallocate routine—it deallocates its own memory.

Program 1: DOS Memory Allocation

Note: This source code listing is for illustrative purposes only. It requires an assembler to enter.

```
                        page 50,132
 0000                   alloc segment para
                            assume cs:alloc
                            assume ds:alloc
                            assume es:alloc
 0000                   allocate proc far
                        ;
                        ;Routine to allow BASIC to make DOS
                         call to allocate a block of memory
                        ;outside of BASIC's own segment. CALL
                         ALLOC(MEMORY)-when BASIC calls the
                        ;routine, MEMORY contains the number
                         of bytes to be allocated.  When the
                        ;routine returns to BASIC, MEMORY
                         contains the segment address of the
                        ;allocated block of memory.  A 7 or 8
                         indicates allocation failed.
                        ;
 0000  55               push bp
 0001  8B EC            mov bp,sp
 0003  8B 5E 06         mov bx,[bp+6]   ;get address of MEMORY
 0006  8B 1F            mov bx,[bx]   ;get number of bytes to
                         be allocated
 0008  B4 48            mov ah,48h  ;DOS function number
 000A  CD 21            int 21h   ;DOS call itself
 000C  8B 5E 06         mov bx,[bp+6]   ;address of MEMORY
 000F  89 07            mov [bx],ax   ;put segment address of
                         allocated memory in MEMORY
 0011  5D               pop bp
 0012  CA 0002          ret 2
                        ;
 0015                   allocate endp
 0015                   alloc ends
                        end
```

Program 2: DOS Memory Deallocation

Note: This source code listing is for illustrative purposes only. It requires an assembler to enter.

```
                        page 50,132
 0000                   dealloc segment para
                            assume cs:dealloc
                            assume ds:dealloc
                            assume es:dealloc
 0000                   dlc proc far
                        ;
                        ;Routine to allow BASIC to make DOS
                         call to deallocate a block of memory
                        ;Previously allocated using ALLOC. CALL
                         DEALLOC(MEMORY)-when BASIC calls the
                        ;routine, MEMORY contains the segment
                         address of the block of memory to be
                        ;dealloc.  When the routine returns to
                         BASIC, MEMORY contains either the
                        ;original segment address or an error
                         code.  A 7 or 9 indicates allocation
                         failed.
                        ;
 0000  55               push bp
 0001  06               push es
 0002  8B EC            mov bp,sp
 0004  8B 5E 06         mov bx,[bp+6]   ;get address of MEMORY
 0007  8E 07            mov es,[bx]   ;get segment address of
                         block to be deallocated
 0009  B4 49            mov ah,49h  ;DOS function number
 000B  CD 21            int 21h  ;DOS call itself
 000D  8B 5E 06         mov bx,[bp+6]
 0010  89 07            mov [bx],ax   ;put error code in MEMORY
 0012  07               pop es
 0013  5D               pop bp
 0014  CA 0002          ret 2
                        ;
 0017                   dlc endp
 0017                   dealloc ends
                        end
```

The Honor System

After studying Program 3, perhaps you've noticed another good reason for BASIC programmers to have access to these DOS calls: It's possible to put a machine language subroutine outside BASIC's 64K memory area, thus saving some space for BASIC programs. Better yet, you don't have to worry about where in memory you're hiding the routine—DOS takes care of it. If you use a lot of machine language subroutines, or store large amounts of data in memory, you'll have a lot more room to work with if you don't have to put everything inside BASIC's own segment.

One final comment about the DOS memory allocation functions: Please use them. Think of it as an honor system. If everyone relies on DOS to determine where their programs reside in memory, we can all feel confident that our coresident programs are not overlapping and conflicting with each other. But if too many programmers bypass these DOS functions, the rest of us won't dare to rely on them, either. After all, DOS can protect only the data or programs that it knows about.

Program 3: DOS Memory Functions in BASIC

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
IL 10 CLEAR ,&HFFDF:REM *** Rese
      rve a few bytes just above
       BASIC for alloc. routine
CL 20 DEFINT A-Z
JO 30 DEF SEG:ALLOC=&HFFDF:DMEMO
      RY=2:DEALLOC=0
KD 40 RESTORE 50:FOR X=0 TO 20:R
      EAD Y:POKE X+ALLOC,Y:NEXT:
      REM *** Install alloc.
IJ 50 DATA &h55,&h8b,&hec,&h8b,&
      h5e,&h06,&h8b,&h1f,&hb4,&h
      48,&hcd
EO 60 DATA &h21,&h8b,&h5e,&h06,&
      h89,&h07,&h5d,&hca,&h02,&h
      0
OJ 70 CALL ALLOC(DMEMORY):REM **
      * DOS call to allocate mem
      ory for dealloc. routine
FF 80 DEF SEG=DMEMORY
KH 90 RESTORE 100:FOR X=0 TO 22:
      READ Y:POKE X,Y:NEXT:REM *
      ** Install dealloc.
OE 100 DATA &h55,&h06,&h8b,&hec,
       &h85,&h5e,&h06,&h8e,&h07,
       &hb4,&h49,&hcd
HN 110 DATA &h21,&h8b,&h5e,&h06,
       &h89,&h07,&h07,&h5d,&hca,
       &h02,&h00
PL 120 CALL DEALLOC(DMEMORY)
LA 130 END                     ©
```

Meet ED The AmigaDOS Editor

Christopher J. Flynn

AmigaDOS—the command-driven operating system which underlies the graphics-oriented Workbench—contains two text editors. Although they aren't full-fledged word processors, these editors are ideal for entering program source code, creating batch files, and even writing short documents. This article shows how to use ED, the more powerful of the two editors. For more information on AmigaDOS and batch files, see "Introduction To AmigaDOS," a two-part series in the January and February 1986 issues of COMPUTE!, *and "AmigaDOS Batch Files," April 1986.*

The Amiga comes with more software than most people realize. Besides Amiga BASIC, Electronic Arts' *Kaleidoscope*, Mindscape's *Amiga Tutor*, the RAM disk, the speech synthesizer, the printer drivers, the icon editor, the calculator, the clock, and numerous demo programs, there are also three complete text editors. Most people know about the Notepad because it's available from the Workbench. But the other two text editors—ED and EDIT—don't show up as icons and must be run from an AmigaDOS CLI (Command Line Interface) window.

The most powerful of these text editors is ED. Although it doesn't handle multiple fonts and styles like the Notepad, it has many more editing functions and is the ideal tool for writing AmigaDOS batch files or program source code. EDIT, on the other hand, is a little more specialized. It is a sequential file editor. In practice, EDIT is best used to make changes to an existing disk file. You'll probably prefer to use ED for composing new text.

We'll be exploring ED version 1.10. Future releases of ED may change things around a little and introduce new features, so keep this in mind.

Starting ED

Where is ED hiding? Even if you peek through every nook and cranny of the Workbench, you will not find an icon for ED. It turns out that ED is actually an AmigaDOS command. This means that you have to start ED from a CLI window.

If you've never used a CLI window before, your first step will be to activate the CLI. Open the Workbench and check the contents of the System drawer. If CLIs are activated, you'll see a cube-shaped icon labeled CLI in this drawer. If the icon is not present, point to the Preferences icon and double-click the mouse's left button. Look for the CLI On/Off selector on the Preferences screen and click on the On box, then exit Preferences by specifying Save (*not* Use). Now when you reopen the System drawer, it should contain a CLI icon. If not, go back to Preferences and make sure CLI is turned on. (If you find yourself using the CLI often, you may want to drag the CLI icon from the System drawer into the main Workbench window to avoid the extra step of opening the System drawer.) To open a CLI window, double-click on the CLI icon. Now you'll have a window in which you can type AmigaDOS commands.

ED can be started in two ways:

ED *filename* [SIZE *n*]

RUN ED *filename* [SIZE *n*]

The first method starts ED from the CLI which you've just activated. It ties up the CLI until you're finished with ED. In other words, you have to leave ED before issuing other AmigaDOS commands. When you specify RUN ED, AmigaDOS automatically starts another CLI task for you and starts ED in this new CLI. Thus, you can temporarily suspend ED by moving the mouse to another window. You can go back to the original CLI and issue other AmigaDOS commands. If you are adventuresome, you can even have multiple ED sessions in progress at the same time. (What you're really doing is multitasking more than one AmigaDOS simultaneously.)

In either case, the ED command requires a filename. You can either supply the name of an existing disk file you wish to edit, or create a new file by specifying a new filename. Remember that Amiga filenames can be up to 30 characters long. So, choose filenames that take advantage of this feature. It helps you recognize your files later on.

There is a SIZE option for the ED command. (Don't type in the brackets, by the way. Brackets just signify options.) A text document must be able to fit entirely in memory. ED just cannot handle a document partly on disk and partly in memory. The SIZE option gives you a way of telling ED how much memory you want to set aside for working on the document. If you don't type in SIZE, ED will set aside 40K for you. The maximum SIZE is determined by the amount of memory you have.

Here are a few examples of commands for starting up ED:

ED GROCERY-LIST

ED WAR-PEACE-BOOK-REPORT SIZE 90000

When SIZE is used, type out the number. Note that 90,000 bytes is typed as 90000 and not as 90,000 or 90K.

Leaving ED

When ED has been successfully started, its display occupies the entire screen. So, how can you return to the CLI? There is no close gadget on ED's window. There is nothing to point at and click. Instead, ED requires either a Quit or an Exit command. Press the ESC (escape) key. An asterisk appears on the last line of the display. Type either Q for Quit or X for Exit and then RETURN. That's all there is to it.

There is a difference between Quit and Exit. Q leaves the editor without saving the document to disk. Anything you have typed will be lost. ED recognizes that this can be quite an inconvenience, so if you do type Q, ED displays the following warning message:

Edits will be lost - type Y to confirm:

Pressing Y at this point gets you out of ED, and no text is saved. If you type anything else, ED lets you continue working on your document.

ESC-X, the Exit option, *does* save the document on disk, using the filename you specified when you started ED. No messages are given. When ED finishes, you're back in the CLI and can then proceed with other AmigaDOS commands. When you're finished with the CLI, type ENDCLI. If you've got only one CLI window running, this returns you to the Workbench.

ED Commands

There are two types of editor commands in ED. The more direct ones are called *immediate commands* because you can enter them while typing text. Examples are line insertions and deletions. Immediate commands are always CTRL key combinations. The other category—*extended commands*—can be typed only when in the command mode. ESC-Q and ESC-X are examples. Pressing ESC opens the lowest display line on your screen for these extended commands.

When ED starts, it positions the cursor at the upper-left corner of the screen. If you are working on a new document, the screen is blank. Otherwise, the screen shows the first page of the document.

If you're creating a new document, just start typing. Notice what happens when the text approaches the right side of the screen. If a word is too long to fit on the remainder of the line, ED pulls the word down to the next line. You can keep typing without being concerned about hitting RETURN at the end of a line as you would on a typewriter.

There are several ways of correcting typos. The BACK SPACE key deletes the character to the left of the cursor. DEL deletes the character under the cursor. Table 1 lists other ways of deleting text.

ED is a full-screen editor, so you can move the cursor wherever you want with the arrow keys. To insert text, position the cursor at the desired location and begin typing. Notice that ED does not have a strikeover mode. Unwanted text has to be deleted—you can't just type over it.

The Insertion Gotcha

Try typing a few fairly long lines. Now, move the cursor to the beginning of the text. Start typing again. The existing text on the current line is moved to the right off the edge of the screen. During insertions, ED neither brings the excess text down to the next line nor enforces margins.

The disappearing text is not lost, however. ED has made one long line. The long line can be split at any point by placing the cursor where you want and pressing RETURN. If you're working with ordinary text, not source code or batch files, this may leave gaps of several spaces between sentences. To clean up the appearance, the extra spaces will have to be removed. Some other lines may need adjusting as well.

Using The Extended Commands

Extended commands (Table 2) can be typed only when ED is in the extended command mode, entered by pressing the ESC key. The cursor appears on the last line of the display. At this point, you can type one or more extended commands. It's quite handy to be able to give ED a series of commands separated by semicolons (;). When you press

Table 1: ED Immediate Commands

Command	Description
Special Keys	
BACK SPACE	Deletes the character to the left of the cursor.
DEL	Deletes the character under the cursor.
ESC	Switches to extended command mode.
RETURN	Ends the line at the cursor and starts a new line.
TAB	Moves the cursor right, adding spaces, to the next tab position.
up-arrow	Moves the cursor up one line.
down-arrow	Moves the cursor down one line.
left-arrow	Moves the cursor one character position to the left.
right-arrow	Moves the cursor one character position to the right.
Control Key Combinations	
CTRL-A	Inserts a line after the line on which the cursor is located.
CTRL-B	Deletes the line on which the cursor is located.
CTRL-D	Scrolls the text down 12 lines toward the beginning of the document.
CTRL-E	If the cursor is at the top of the screen, moves the cursor to the bottom of the screen. If the cursor is at the bottom of the screen, moves the cursor to the top of the screen.
CTRL-F	Switches the case (upper to lower or lower to upper) of the character under the cursor.
CTRL-G	Repeats the last extended command which was issued.
CTRL-H	Deletes the character to the left of the cursor. Equivalent to the BACK SPACE key.
CTRL-I	Moves the cursor right to the next tab position. Equivalent to the TAB key.
CTRL-M	Equivalent to the RETURN key.
CTRL-O	If the cursor is on a nonblank character, deletes all characters from the cursor to the first space. If the cursor is on a space, deletes all spaces from the cursor to the first nonblank character.
CTRL-R	Moves the cursor left to the first space after previous word on the current line.
CTRL-T	Moves the cursor right to the first character of the next word on the current line.
CTRL-U	Scrolls the text up 12 lines toward the end of the document.
CTRL-V	Redisplays (Verifies) the screen. Insures that all the text is visible and is useful after moving or sizing the display window.
CTRL-Y	Deletes all characters on the line starting with the character under the cursor.
CTRL-[	Switches to the extended command mode. Equivalent to the ESC key.
CTRL-]	If the cursor is at the start of the line, moves the cursor to the end of the line. If the cursor is at the end of the line, moves the cursor to the start of the line.

RETURN, ED acts on the command or commands you've requested.

Extended commands can move the cursor, mark blocks of text for certain operations, and perform searches and exchanges. Some of the operations are tricky and require care. Cursor commands apply only to the cursor position in the text and not to the command line. This is fine except that you can't see the cursor in the text. You have to remember where the cursor is before you use some of the extended commands.

Sections of text can be marked by block start (BS) and block end (BE) commands. Blocks can be deleted, copied elsewhere in the document, or saved to disk. Marking a block involves moving the cursor to the first line in the block and executing the BS extended command. The end of the block is marked similarly with BE. Unfortunately, there is no visible indication of the defined text. Be very careful of cursor movements. The only help ED offers is the show (SH) command. It displays the first and last line of the block and some other information.

Text search and exchange operations work without a hitch. You can search forward (F) or backward (BF) through the document. You can exchange (E or EQ) one text string for another. Lowercase text can be treated as matching uppercase text (UC), or it can be treated as not matching (LC).

The repeat (RP) command is often used for exchanges. RP causes the command following it to be executed repeatedly until something (an error, for example) stops it. Thus, RP E carries out multiple exchange operations. Here is an example:

T; RP E /Compute/COMPUTE/

Here, the typing of COMPUTE! is being corrected. T moves the cursor to the top of the document so that the entire document will be examined. RP precedes the exchange command. Note that the two text strings are delimited by slashes. This is ED's convention when text strings are used. A "Search failed" error occurs when *Compute* can no longer be found in the text. This halts the repeat command, and the entire document will have been corrected.

The save command (SA) saves the document to disk without exiting ED. You should do this periodically to prevent disasters in the event of a power failure.

Overall, ED is an excellent general-purpose text editor. You can use it when programming, since it works with any language that accepts ASCII text files as input (including Amiga BASIC). ED can also prepare data files or help you write short letters and notes. It's not a fancy word processor, but it can handle smaller, less complex tasks quite well. ©

Table 2: ED Extended Commands

Note: /s/ refers to a single text string (/this is a string/).
/s/t/ refers to two text strings (/brown/blue/).

Command	Description
A /s/	Inserts the string on a new line after the current line.
B	Moves the cursor to the end (bottom) of the document.
BE	Places an end-of-block marker at the cursor.
BF /s/	Searches the document for the string going in a direction from the cursor toward the beginning of the document (backward find).
BS	Places a start-of-block marker at the cursor.
CE	Moves the cursor to the end of the current line.
CL	Moves the cursor one character position to the left.
CR	Moves the cursor one character position to the right.
CS	Moves the cursor to the start of the line.
D	Deletes the current line. Moves all following lines up.
DB	Deletes the text marked by start-block and end-block markers.
DC	Deletes the character at the current cursor position.
E /s/t/	Replaces (exchanges) occurrences of the first string with the second string.
EQ /s/t/	The same as E, but asks you to confirm the replacement each time a match is found. Type Y or N in response to the Exchange ? prompt.
EX	Extends the right margin allowing additional text to be typed.
F /s/	Searches the document for the string going in a direction from the cursor position toward the end of the document (find).
I /s/	Inserts the string on a new line before the current line.
IB	Inserts the block of text marked by start-block and end-block markers after the current line.
IF /s/	Inserts the contents of a file before the current line. The filename is given by /s/.
J	Joins the current line with the next line. This makes one new line where there were formerly two.
LC	Treats upper- and lowercase characters as different in searches.
M *n*	Moves the cursor to the line number given by *n*.
N	Moves the cursor to the starting position of the next line.
P	Moves the cursor to the starting position of the previous line.
Q	Quits ED without first saving the text. A warning message will be given stating that the text may be lost.
RP	Repeats commands. Commands are typed following RP. For example, T; RP E /brown/red/ moves the cursor to the top of the document. The Exchange command is repeated, thus changing all occurrences of *brown* to *red*. Repeat ends when an error is found. In this case, an error occurs after all the changes have been made since *brown* can no longer be found.
S	Splits the current line at the cursor location. This makes two lines where there was formerly one.
SA	Saves the document to the file specified by the original ED command. Use SA periodically to make sure you have a good copy of the text on disk.
SB	Shows the text block marked by start-block and end-block markers. The block (and any following text) will be displayed starting at the top of the screen.
SH	Shows the filename, tab distance, margin settings, first and last line of any marked text block, and the buffer full percentage.
SL *n*	Sets the left margin to the position specified by *n*. SL affects the margin setting for the entire document. New text will be typed within the margins. Existing text is not automatically reformatted when the margins change.
SR *n*	Sets the right margin to the position specified by *n*. SR affects the margin setting for the entire document. New text will be typed within the margins. Existing text is not automatically reformatted when the margins change.
ST *n*	Sets the distance the cursor moves when the TAB key is pressed.
T	Moves the cursor to the top of the document.
U	Undoes any changes made to the current line. This does not restore line deletes (D). It also does not work if you have moved the cursor from the current line.
UC	Treats upper- and lowercase characters as equivalent for searches (for example, *A* will match *a*).
WB /s/	Writes the text block marked by start-block and end-block markers to the file specified by /s/.
X	Exits ED first making sure that the document has been saved on disk.

Converting IBM ML To BASIC DATA

Mark Russinovich With Dennis Moul

This short utility converts object code created with a machine language assembler into DATA statements ready to be merged with a BASIC program. It works on any IBM PC, PCjr, or compatible with DOS 2.0 or higher.

An efficient way of speeding up crucial parts of BASIC programs or performing operations not possible in BASIC is to write a machine language subroutine. Usually, the machine language (ML) routine is loaded from disk by the BASIC program or is encoded in BASIC DATA statements that are POKEd into memory. The latter method has the advantage of making the BASIC program a stand-alone unit, not dependent on other files that must be on the same disk. Its major disadvantage, though, is that if the ML routine is more than a few bytes long, the job of converting the object code to DATA is extremely tedious and error-prone. One minor mistake could mess up the whole routine and possibly crash the system.

The solution is Program 1 below, "BIN2DAT." It takes an ML (binary) file on disk and converts it to DATA statements, ready to be merged into your BASIC program. It is impeccably reliable and takes only seconds to do its work.

Using BIN2DAT

After typing in Program 1 and saving it on disk, make sure that the ML object file you wish to convert into DATA statements is stored on disk in the .COM format. This is necessary because .EXE files have relocation information used by DOS when they are loaded into memory. Since DOS isn't used when a BASIC program POKEs an ML routine into memory, an .EXE program would not be relocated and therefore would not execute. If you've already written an .EXE file that you wish to convert to DATA statements, convert it to .COM format by using the EXE2BIN program included on the PC-DOS disk.

Now follow these steps:

1. Run BIN2DAT. It asks you for the filename of the .COM file you wish to convert. Enter the filename and press Enter. As an extra safeguard, BIN2DAT makes sure that the file has a .COM extension.

2. BIN2DAT prompts you for the output filename (the file that will contain the DATA statements). If you simply press Enter here, the filename defaults to the one displayed within brackets.

3. Next, you're asked for the starting line number of the DATA statements. Again, a default, which is line 100, is printed within brackets. Either press Enter or type your own starting line number.

4. BIN2DAT now asks for the line number increment (the default is 10) and the numeric base of the data—decimal or hexadecimal. The base makes little difference, but the default is hexadecimal because sometimes it's useful to compare the .LST file generated by the assembler with the DATA statements.

Merging The DATA

Once you've entered all the required information, BIN2DAT creates the BASIC data file to your specifications. To merge it with your BASIC program, load the BASIC program and type:

MERGE *"filename.ext"*

You'll notice that the first line in the file has only one data value. This isn't part of the ML. This value is the size of the ML routine in bytes, minus one. Therefore, it corresponds to the upper limit of a FOR-NEXT loop that is required to POKE the ML routine into memory.

Next are the lines containing the data for the ML program. An example of an ML routine is seen in Program 2, "EXAMPLE.ML." Program 3, "Demo DATA," shows the file produced by BIN2DAT after converting the .COM file produced by an assembler and EXAMPLE.ML. Extra lines have been added to POKE the ML routine into memory and CALL it. Examining these listings should clear up any questions about how to use BIN2DAT.

How It Works

BIN2DAT is fairly straightforward. Once all the information has been entered by the user, the SHELL command is used to create a file with the directory entry of the ML file. SHELL allows the use of DOS commands from BASIC, but in the DOS 2.0/2.1 generation, it has the flaw of altering memory locations 30H and 31H, which happen to point to the beginning of the BASIC program in memory. To overcome this, the values for these locations are PEEKed before the SHELL command is executed and then POKEd back later.

The next part of the program reads the size of the ML file out of the directory random file which was made by SHELL. Then it begins constructing the DATA statements, which are sent to the output file. The first DATA line has only the count value (described above). Subsequent lines have ten data numbers each. The MOD 10 function checks for the end of a line. When a line ends, it is sent to the output file and a new line is started.

After the ML program has been completely read and the new file is finished, the CLOSE command closes the input and output files, and the program terminates.

Several changes can make BIN2DAT serve your particular needs better. If you usually start your data on some line other than 100, this default value can be changed. Also, the default values for the line increment and numeric base can be changed to make running the program easier. If you want to have more than or fewer than ten items per data line, you can change the number 10 in each MOD function to some other number.

Program 2: EXAMPLE.ML

Note: This source code listing is for illustrative purposes only. It requires an assembler to enter.

```
; This is a sample assembly
; language program that will
; be poked in and run from
; BASIC.

prog    segment

        assume  cs:prog,ds:prog

main    proc    far
        push    ax
        push    bx
        push    dx

; Print characters

        mov     bx,offset mess1
print:  mov     dl,cs:[bx] ;get char
        cmp     dl,Ø       ;are we through?
        je      exit       ;yes, return
        inc     bx         ;no, get nxt char
        mov     ah,2       ;dos print routine
        int     21h
        jmp     print      ;get more
exit:   pop     dx         ;restore stack
        pop     bx
        pop     ax
        ret

mess1   db Ødh,Øah,'This is output of a'
        db ' sample assembly language'
        db ' program.',Ødh,Øah,Ødh,Øah
        db Ødh,Ø

main    endp

prog    ends
        end     main
```

Program 1: BIN2DAT

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
LJ 10 DEF SEG
BH 20 KEY OFF
GO 30 ON ERROR GOTO 570
OM 40 :
CF 50 REM Print title and get in
      fo
OO 60 :
KN 70 PRINT "Binary to Data Stat
      ement Converter"
NM 80 PRINT "(c) Copyright 1986,
         Compute! Publications"
FL 90 PRINT
OD 100 INPUT "File to convert: "
      ,FSOURCE$
BF 110 IF INSTR(FSOURCE$,".COM")
       =Ø AND INSTR(FSOURCE$,".c
       om")=Ø THEN  PRINT:BEEP:P
       RINT "File must have .COM
        extension.":END
OO 120 FILEN=INSTR(FSOURCE$,".")
       -1:FILEN$=LEFT$(FSOURCE$,
       FILEN)+".BAS"
PL 130 PRINT "Data file [";FILEN
       $;"]";:INPUT ": ",FDEST$
NF 140 IF FDEST$="" THEN FDEST$=
       FILEN$
HF 150 INPUT "Starting line numb
       er [100]: ",SLN
ED 160 INPUT "Line increment [10
       ]: ",LINC
JF 170 INPUT "Hex/decimal [h]: "
       ,H$
FD 180 IF SLN=Ø THEN SLN=100
HC 190 IF LINC=Ø THEN LINC=10
GM 200 IF H$="" OR H$="h" OR H$=
       "H" THEN H=1
MG 210 :
NG 220 REM Capture directory in
       random file
NK 230 :
JA 240 P1=PEEK(&H3Ø):P2=PEEK(&H3
       1)
NC 250 SHELL "dir "+FSOURCE$+" >
       $$zztemp"
MG 260 POKE &H3Ø,P1:POKE &H31,P2
DB 270 OPEN "$$zztemp" FOR INPUT
        AS 2
OJ 280 FOR I=1 TO 4:INPUT#2,DMY$
       :NEXT
JH 290 INPUT#2,ENTRY$
PL 300 REM Get size of com file
        from dir
KO 310 SIZE=VAL(MID$(ENTRY$,16,6
       ))
ND 320 CLOSE #2 :KILL "$$zztemp"
NL 330 :
EL 340 REM Open com file and new
        dat file
NP 350 :
NF 360 OPEN FSOURCE$ AS 1 LEN=1
MD 370 FIELD 1,1 AS BYTE$
IF 380 OPEN FDEST$ FOR OUTPUT AS
        2
LO 390 LINNUM=SLN+LINC
HF 400 LIN$=STR$(SLN)+" DATA"
```

```
IE 410 IF H=1 THEN LIN$=LIN$+" &
       h"+HEX$(SIZE-1) ELSE LIN$
       =LIN$+STR$(SIZE-1)
MH 420 PRINT#2,LIN$
PJ 430 LIN$=STR$(LINNUM)+" DATA
       "
CM 440 LINNUM=LINNUM+LINC
EH 450 FOR COUNT=1 TO SIZE
BH 460   GET #1,COUNT
OM 470   WBYTE$=BYTE$
CC 480   IF H=1 THEN NUM$="&h"+H
       EX$(ASC(WBYTE$)) ELSE NUM
       $=STR$(ASC(WBYTE$)):
              NUM$=RIGHT$(NUM$,L
       EN(NUM$)-1)
LI 490   IF COUNT MOD 10>0 THEN
       520
DB 500   PRINT#2,LIN$+NUM$:LIN$=
       STR$(LINNUM)+" DATA "
PC 510   LINNUM=LINNUM+LINC:GOTO
        530
MF 520   LIN$=LIN$+NUM$+","
NH 530 NEXT
HP 540 IF COUNT MOD 10<>1 THEN L
       IN$=LEFT$(LIN$,LEN(LIN$)-
       1):PRINT#2,LIN$
PE 550 CLOSE
FD 560 PRINT:PRINT "File written
       ":END
PO 570 BEEP:PRINT "DOS error - a
       borting.":CLOSE:END
```

Program 3: Demo DATA

For instructions on entering this listing, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

```
GC 10 REM  This program will pok
      e in an
HF 20 REM  assembly language pro
      gram and
LL 30 REM  then CALL it.
OM 40 :
ND 50 DEF SEG=&H1700
ON 60 READ COUNT
EB 70 FOR MEM=0 TO COUNT
JP 80   READ BYTE
LB 90   POKE MEM,BYTE
NN 100 NEXT
MF 110 :
KC 120 SAMPLE=0
OI 130 CALL SAMPLE
LC 140 END
NN 150 :
CM 160 DATA &h55
OP 170 DATA &h50,&h53,&h52,&hBB,
       &h19,&h0,&h2E,&h8A,&h17,&
       h80
LM 180 DATA &hFA,&h0,&h74,&h7,&h
       43,&hB4,&h2,&hCD,&h21,&hE
       B
HG 190 DATA &hF1,&h5A,&h5B,&h58,
       &hCB,&hD,&hA,&h54,&h68,&h
       69
KE 200 DATA &h73,&h20,&h69,&h73,
       &h20,&h6F,&h75,&h74,&h70,
       &h75
FO 210 DATA &h74,&h20,&h6F,&h66,
       &h20,&h61,&h20,&h73,&h61,
       &h6D
JM 220 DATA &h70,&h6C,&h65,&h20,
       &h61,&h73,&h73,&h65,&h6D,
       &h62
PG 230 DATA &h6C,&h79,&h20,&h6C,
       &h61,&h6E,&h67,&h75,&h61,
       &h67
DE 240 DATA &h65,&h20,&h70,&h72,
       &h6F,&h67,&h72,&h61,&h6D,
       &h2E
CO 250 DATA &hD,&hA,&hD,&hA,&hD,
       &h0
```

©

64 Fleet List

Buck Childress

Have you ever wished you could zip forward or backward through a program listing at the touch of a key? That capability is especially valuable when you're writing or debugging a long BASIC program. This Commodore 64 utility lets you do exactly that—scroll a program listing up or down on the screen using the 64's special function keys.

"Fleet List" simplifies and speeds up the process of editing a BASIC program listing. As a bonus, it can also tell you the current number of lines in a program and is very easy to use.

Since Fleet List is written entirely in machine language, it must be entered using the "MLX" machine language entry program, published elsewhere in this issue. Be sure you have read and understood the instructions for using MLX before you begin entering the data for Fleet List. When you first run MLX, you'll be asked for starting and ending addresses. The proper values for Fleet List are as follows:

Starting address: C000
Ending address: C367

After you have entered all the data for Fleet List, be sure to use the MLX Save option to store at least one copy of the data before proceeding.

Scroll In Either Direction

To use Fleet List, load it into memory with LOAD *"filename"*,8,1 (for tape, change the ,8,1 to ,1,1), then type NEW and press RETURN to reset memory pointers. Fleet List is now in memory, but it's not active yet. You should first load the BASIC program you wish to edit, then type SYS 49152 and press RETURN to activate Fleet List. (For the utility to function properly, there must be a BASIC program in memory when Fleet List is activated.) It can handle programs up to 1,600 lines in length (a warning is issued if your program is too long).

To scroll the listing forward, press the f1 function key (the text will be dark gray). To scroll backward, press f3 (the text will be black). To move quickly from one part of the program to another, hold down the Commodore key while pressing f1 or f3. You'll see the line numbers spin past on your screen. When you release the Commodore logo key, Fleet List begins listing from that point onward. At other times you may want a slow-motion listing. To slow down the scrolling in either direction, press f2 (SHIFT-f1) or f4 (SHIFT-f3).

If you scroll past the end or beginning of your program, Fleet List simply wraps around to the other end of the program. For instance, say that your program starts with line 10 and ends at line 1000. If you scroll forward past line 1000, Fleet List prints a line on the screen as a marker and then begins to list forward from line 10. If you scroll backward past line 10, Fleet List prints a marker line and begins to work downward from line 1000.

Fleet List also provides an easy way to move immediately to the beginning or end of the program. If you press the f7 key, the list starts at the first line in the program. Remember, Fleet List wraps around the ends of the program automatically, so to get to the very last line, simply scroll backward one line from the beginning.

Line Count

You can find out how many lines you have in your program at any time by pressing the f5 function key. When Fleet List is first activated, it also displays the number of lines in whatever program is currently in memory. As you add and delete lines, the f5 key comes in handy.

Of course, if you're writing a program that uses the function keys for its own purposes, you want to be able to enter them normally in a program line. To allow for this possibility, Fleet List checks for quote and insert modes and does not respond when you're in either mode. When you leave quote or insert mode (usually by pressing RETURN), Fleet List is active again.

Fleet List does not interfere with the process of editing existing program lines, entering new ones, or moving around on the screen with the cursor keys. And there's no need to clear the screen to relist after such activities. Before it begins to list again, Fleet List automatically positions the cursor at the bottom of the screen.

Because Fleet List resides in a memory area that's not normally used by BASIC, you should be able to load and save BASIC programs without disturbing it. However, before loading or saving, it's a good idea to deactivate Fleet List by pressing RUN/STOP-RESTORE. To reactivate Fleet List, type SYS 49152 and press RETURN.

Fleet List

For instructions on entering this listing, please refer to the "MLX" article in this issue of COMPUTE!.

```
CØØØ:AD 14 Ø3 AE 15 Ø3 C9 5E 4F
CØØ8:DØ Ø4 EØ CØ FØ 4F 8D 7Ø 6B
CØ1Ø:C3 8E 71 C3 2Ø BØ C2 8C 57
CØ18:73 C3 8C 74 C3 8C 76 C3 1E
CØ2Ø:8C 78 C3 8C 79 C3 8C 7E B9
CØ28:C3 A9 ØF A2 ØB 8D 2Ø DØ A1
CØ3Ø:8D 21 DØ 8E 86 Ø2 8E 75 92
CØ38:C3 2Ø 44 E5 A9 8E 2Ø D2 25
CØ4Ø:FF A9 Ø8 2Ø D2 FF AØ ØØ Ø7
CØ48:2Ø FB CØ 2Ø 18 C3 A9 5E 74
CØ5Ø:8D 7E C3 A2 CØ 78 8D 14 F1
CØ58:Ø3 8E 15 Ø3 58 6Ø AD 74 E5
CØ6Ø:C3 DØ 6D A5 D4 DØ 66 A5 5C
CØ68:D8 DØ 62 AD 8D Ø2 C9 Ø3 BC
CØ7Ø:BØ 5E A5 CB C9 4Ø FØ 58 DC
CØ78:C9 Ø1 DØ Ø3 8D 79 C3 C9 ØD
CØ8Ø:Ø3 DØ ØC 8D 74 C3 2Ø BØ B5
CØ88:C2 8C 76 C3 4C A3 C2 C9 D9
CØ9Ø:Ø6 DØ 32 8D 74 C3 8D 75 AB
CØ98:C3 AØ ØØ 8C 7E C3 A5 FB 37
CØAØ:48 A5 FC 48 A9 Ø1 2Ø FB 61
CØA8:CØ 68 85 FC 68 85 FB A2 19
CØBØ:18 86 CC 8E 7E C3 AØ ØØ A6
CØB8:18 2Ø FØ FF 2Ø 4D C3 2Ø 4A
CØCØ:18 C3 4C A3 C2 C9 Ø4 9Ø D8
CØC8:Ø4 C9 Ø6 9Ø Ø6 8D 78 C3 A3
CØDØ:6C 7Ø C3 8D 74 C3 AD 78 7C
CØD8:C3 FØ 13 A9 ØØ 8D 73 C3 56
CØEØ:8D 78 C3 A2 18 AØ ØØ 18 45
CØE8:2Ø FØ FF 2Ø 4D C3 AC 73 FE
CØFØ:C3 DØ Ø5 AD 79 C3 DØ Ø3 83
CØF8:4C 7B C1 8C 7C C3 8C 7D ØA
C1ØØ:C3 A5 FD 48 A5 FE 48 2Ø EC
C1Ø8:BØ C2 AØ Ø1 84 CC 88 B1 D2
C11Ø:FB FØ Ø6 2Ø C4 C2 4C ØA 64
C118:C1 2Ø C4 C2 2Ø C4 C2 B1 94
C12Ø:FB FØ 1E 2Ø C4 C2 A5 FB 1C
C128:A6 FC 91 FD 2Ø CB C2 8A 9Ø
C13Ø:91 FD 2Ø CB C2 2Ø E9 C2 E9
C138:2Ø C4 C2 2Ø C4 C2 4C ØA 2B
C14Ø:C1 2Ø F2 C2 A5 FD A6 FE A8
C148:8D 7A C3 8E 7B C3 AØ ØØ BE
C15Ø:A5 FE C9 DØ BØ 16 E6 FD 56
C158:DØ Ø8 E6 FE A5 FE C9 DØ AØ
C16Ø:BØ ØA A5 FE 85 CC 98 91 85
C168:FD 4C 56 C1 2Ø BØ C2 68 96
C17Ø:85 FE 68 85 FD AD 7E C3 43
C178:DØ Ø1 6Ø AØ Ø1 84 CC 88 F6
C18Ø:AD 74 C3 C9 Ø4 FØ 2B 8C D3
C188:86 Ø2 AD 76 C3 C9 Ø2 DØ Ø7
C19Ø:Ø6 2Ø F2 C2 2Ø F2 C2 A9 A5
C198:Ø1 8D 76 C3 2Ø F2 C2 2Ø 7D
C1AØ:F2 C2 AØ Ø1 B1 FD FØ F4 CE
C1A8:85 FC 88 B1 FD 85 FB 4C A4
C1BØ:E2 C1 A9 ØB 8D 86 Ø2 AD 34
C1B8:76 C3 C9 Ø1 DØ Ø6 2Ø CB 5C
C1CØ:C2 2Ø CB C2 A9 Ø2 8D 76 3A
C1C8:C3 B1 FD 85 FB 2Ø CB C2 6D
C1DØ:B1 FD DØ Ø9 2Ø BØ C2 2Ø CØ
C1D8:13 C3 4C C9 C1 85 FC 2Ø 3B
C1EØ:CB C2 AD 8D Ø2 48 48 8D D8
C1E8:77 C3 C9 Ø2 DØ Ø8 2Ø 4D A6
C1FØ:C3 A9 91 2Ø D2 FF B1 FB EA
C1F8:AA 2Ø C4 C2 B1 FB 2Ø CD 2A
C2ØØ:BD 2Ø C4 C2 A9 2Ø 2Ø D2 12
C2Ø8:FF 68 AA EØ Ø2 DØ Ø8 A9 18
C21Ø:2Ø 2Ø D2 FF 2Ø D2 FF AØ F4
C218:ØØ B1 FB FØ 3D EØ Ø2 DØ DA
C22Ø:Ø6 2Ø C4 C2 4C 17 C2 C9 83
C228:8Ø 9Ø 26 A6 D4 DØ 22 38 A7
C23Ø:E9 7F AA AØ FF CA FØ Ø8 FE
C238:C8 B9 9E AØ 1Ø FA 3Ø F5 31
C24Ø:C8 B9 9E AØ C9 8Ø BØ Ø6 2E
C248:2Ø D2 FF 4C 4Ø C2 38 E9 BE
C25Ø:8Ø 2Ø D2 FF 2Ø C4 C2 4C 5E
C258:17 C2 68 C9 Ø2 FØ Ø3 2Ø BD
C26Ø:4D C3 A5 CB C9 Ø4 9Ø 31 9F
C268:C9 Ø6 BØ 2D 8D 74 C3 AD BØ
C27Ø:8D Ø2 C9 Ø2 FØ ØA AD 77 19
C278:C3 C9 Ø2 DØ Ø6 2Ø 4D C3 AE
C28Ø:4C 7B C1 AD 8D Ø2 FØ F8 6D
C288:C9 Ø2 BØ F7 A2 ØØ AØ ØØ 5F
C29Ø:E8 DØ FD C8 DØ FA 4C 7B 91
C298:C1 AD 77 C3 C9 Ø2 DØ Ø3 9Ø
C2AØ:2Ø 4D C3 A9 ØØ 85 C6 8D CD
C2A8:74 C3 8D 79 C3 6C 7Ø C3 17
C2BØ:AØ ØØ A2 8Ø 84 FB 86 FD Ø2
C2B8:A9 Ø8 A2 C3 8D 73 C3 85 EC
C2CØ:FC 86 FE 6Ø E6 FB DØ Ø2 17
C2C8:E6 FC 6Ø E6 FD DØ FB E6 8D
C2DØ:FE A5 FE C9 DØ 9Ø F3 2Ø 8C
C2D8:44 E5 2Ø 13 C3 A2 Ø8 2Ø Ø8
C2EØ:41 C3 2Ø 3Ø C3 CØ ØØ FØ 11
C2E8:FC EE 7C C3 DØ DC EE 7D C9
C2FØ:C3 6Ø A5 FE C9 C4 BØ Ø6 DD
C2F8:A5 FD C9 81 9Ø ØB C6 FD 5E
C3ØØ:A5 FD C9 FF DØ Ø2 C6 FE 2E
C3Ø8:6Ø AD 7A C3 AE 7B C3 85 27
C31Ø:FD 86 FE A9 ØØ 8D 75 C3 97
C318:2Ø 4D C3 A2 28 A9 2A 2Ø Ø2
C32Ø:D2 FF CA DØ FA AD 75 C3 B4
C328:FØ 23 8E 75 C3 2Ø 4D C3 17
C33Ø:AE 7C C3 AD 7D C3 2Ø CD 8A
C338:BD A2 ØØ 2Ø 41 C3 4C 18 13
C34Ø:C3 BD 55 C3 FØ ØE 2Ø D2 D2
C348:FF E8 4C 41 C3 A9 ØD 2Ø A6
C35Ø:D2 FF AØ ØØ 6Ø 2Ø 4C 49 BA
C358:4E 45 53 ØD ØØ 4F 56 45 C2
C36Ø:52 2Ø ØØ ØØ ØØ ØØ ØØ ØØ 19
```

©

Automatic Typist: Using Apple Exec Files

Mike Miyake

Although it's often overlooked, the EXEC command offers an easy way to extend the power of Applesoft BASIC. EXEC can read and perform commands directly from a disk file, just as if you'd typed the commands on the keyboard yourself. It can also be used as a convenient, built-in merge command for adding frequently used subroutines to Applesoft programs. The example programs below run on any Apple II-series computer; most work with either DOS 3.3 or ProDOS. A disk drive is required.

Have you ever wanted to know the address of a machine language program, or the number code for one of the Apple's 16 low-resolution colors? Are you curious about how a particular Applesoft program uses the computer's memory? In most cases finding the answers to such questions means thumbing through a reference book or typing cumbersome statements like PRINT PEEK (N)+256*PEEK(N+1) to examine memory. And the PEEK statement usually must be typed in immediate mode, since running a short program to get at the information would destroy any program that's already in memory.

"Automatic Typist" shows you how *exec files* can solve such problems. An exec file is simply a text file which you activate with an EXEC command from Applesoft BASIC. It executes like an immediate mode statement—something you type directly on the keyboard, without a line number—but it can also be saved to disk and reused over and over, just like a program. In effect, exec files let you control the computer with disk files that act like immediate commands without disturbing a program that's in memory. We'll show how to put both immediate mode commands and program lines in exec files, and provide some useful examples of what exec files can do.

Creating Exec Files

Type in Program 1, then run it once to make sure it works correctly. Run the program and follow the prompts, entering any filename when prompted. Since this is just for practice, it doesn't matter what filename you use. After that's done, exit the program and type CATALOG to view the new file; it should show up as a text (T) file. Now delete the file (it doesn't contain any data, so you're not losing anything of importance).

Once you're satisfied that Program 1 works correctly, it can be used to create exec files. An exec file ordinarily contains one or more statements in the form of ordinary text. Unfortunately, the Apple II *DOS Manual* tells you very little about how to create such a file. In most cases the simplest way to do so is within a BASIC program. Program 1 illustrates the basic technique. Once the file has been opened (line 18) and a WRITE statement has executed (line 20), all subsequent PRINT and LIST statements send their output to the disk file instead of to the screen. Other BASIC statements function normally while the output of PRINT and LIST is being diverted. When the file is closed (line 1000), PRINT and LIST resume their normal functions.

Program 1 provides you with a template program for creating exec files. It lets you choose a filename, opens the file, and prepares it for writing. To use this program, you need only add appropriate PRINT and/or LIST statements in new lines between lines 20 and 1000 of the template.

Let's try a simple example. Load Program 1, then type in the lines listed in Program 2. The object is to add the lines from Program 2 to the template program. The initial PRINT statements in lines 50–70 write the commands bracketed inside quotes to the disk file. In cases where the exec file itself will contain PRINT statements, it's necessary to write quote characters to the file as well. This is done in line 70 with the variable Q$, which Program 1 defines as CHR$(34) in line 10.

After you finish adding the lines from Program 2, run the program. Enter the filename CC when asked for a filename, then press the space bar when prompted. The text inside quotes in lines 50–70 is written to a disk file named CC. To execute this file, exit the program and type EXEC CC. It displays all 16 lo-res colors in vertical bars on the screen, with a matching number code directly beneath each color bar.

If the CC file doesn't work properly, delete it and repeat the process. If you want to use the same filename (the normal case), it's necessary to delete the old version of a text file before writing an updated version of the file to disk. Unlike

BASIC program files, which automatically replace an existing file with a new file of the same name, text files simply append new information to the end of the existing file.

A Program-Writing Program

The previous example printed immediate mode statements (commands without line numbers) to the exec file. But you can also print numbered program lines to a file. For instance, reload Program 1 and add this line:

```
40 PRINT"100 TEXT:HOME"
```

Run the program and write the file to disk using the filename HOMER. It creates a text file consisting of the BASIC program line 100 TEXT:HOME (we'll explain below why you might want to create this type of file). Although you can write program lines to a disk file with PRINT, it's often more convenient to use LIST instead. One advantage of doing so is that you can type the lines exactly as they normally appear without having to enclose everything in PRINT statements.

To illustrate, let's create the HOMER exec file with LIST instead of PRINT. DELETE the HOMER file from your disk, then reload Program 1 and enter these new lines:

```
21 REM CAPTURE BASIC
22 REM PROGRAM LINES
23 LIST 24,999: GOTO 1000
100 TEXT:HOME
```

The LIST command in line 23 writes every program line from 24 to 999 to the disk file. The GOTO command branches around the data (one line in this case) that we're writing to disk. One disadvantage of this technique is that the lines to be written to disk must fall between 24 and 999, inclusive. By renumbering either the template program or the lines to be written, you should be able to overcome this problem in most cases.

Merging Common Subroutines

Since the EXEC command is analogous to typing, an exec file is a good place to save commonly used subroutines for reuse in different programs. This makes it easy to merge the subroutine contained in the exec file with a program already in memory. To bring the lines into memory, simply type EXEC *filename*. As long as the exec file contains no lines numbered the same as those in the current program, the new lines are added without disturbing the program in memory.

To illustrate, let's save Program 1 in exec file form. Reload Program 1, then add this line:

```
25 LIST 1,20:LIST 1000,
```

Now run the program, entering C.LINES when prompted for a filename. Exit the program, then type NEW, followed by LIST to confirm that no program is in memory. Type EXEC C.LINES and press RETURN. Several bracket prompts will scroll past as the computer enters each program line automatically. When the cursor reappears, type LIST. Program 1 is back in memory, just as if you had typed each line manually.

It's not difficult to see how much programming time this method could save, particularly if you build up a library of commonly used subroutines that each use different ranges of line numbers.

Last BLOAD

Program 3 is an exec file that comes in handy in many different situations. Its purpose is to tell you the load address and length of the last file that was BLOADed into memory. Knowing this information lets you run machine language programs immediately with a CALL statement, or copy them without using DOS 3.3's FID utility. Unlike the other examples presented here, this one is for DOS 3.3 only—it does not work with ProDOS.

The procedure for creating this file should be familiar by now: Reload Program 1, add the lines listed in Program 3, then run the program. This exec file uses pointer locations applicable to a 48K Apple II. If you have a 16K or 32K system, change the pointer locations as shown here:

```
32K: address=PEEK(27250)+256* PEEK(27251)
     length=PEEK(27232)+256* PEEK(27233)

16K: address=PEEK(10866)+256* PEEK(10867)
     length=PEEK(10848)+256* PEEK(10849)
```

Memory Map

Program 4 contains the lines to add to Program 1 to create another useful exec file. This one shows the memory locations of the current BASIC program and its strings and variables. To use it, load and run the BASIC program you're curious about, then type EXEC *filename* in immediate mode. The pointer locations used by this file are discussed on page 140 of the *Applesoft II BASIC Reference Manual.*

For instructions on entering these listings, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

Program 1: Exec File Maker

```
8B 10 D$ = CHR$ (4):Q$ = CHR$ (3
      4)
55 11 TEXT : HOME : HTAB 15: PRI
      NT "MAKE FILES:"
69 12 VTAB 6: INPUT "FILENAME: "
      ;N$: IF LEN (N$) = 0 THEN
      12
D2 14 HTAB 1: VTAB 8: CALL - 958
FE 16 PRINT "INSERT NON-WRITE PR
      OTECTED DISK": PRINT "AND
      PRESS SPACE BAR WHEN READY
      .": PRINT : PRINT "  => ";
      : GET A$
9C 18 PRINT D$"OPEN"N$: PRINT D$
      "CLOSE"
E5 20 PRINT D$"OPEN"N$: PRINT D$
      "WRITE"N$
61 1000 PRINT D$"CLOSE"
B6 1002 VTAB 14: PRINT "DO IT AG
        AIN?";: GET A$
8D 1004 IF A$ = "Y" THEN 14
F2 1006 END
```

Program 2: Color Chart

```
49 REM  CC.LINES
50 PRINT "TEXT:GR"
60 PRINT "FORI=1TO15:COLOR=I:VL
   IN20,39 AT2*I:VLIN20,39 AT 2
   *I+1:NEXT"
70 PRINT "PRINT"Q$"0   2   4
   6   8   10  12  14"Q$":PRINT
   "Q$"  1   3   5   7   9   11
     13  15"Q$
```

Program 3: Last BLOAD

```
49 REM  LAST BLOAD / 3.3 DOS/48
   K
50 PRINT "TEXT:HOME"
100 PRINT "PRINT"Q$"LAST BLOAD"
    Q$":PRINT:PRINT"Q$"M/L FILE
    : ADDRESS = "Q$"PEEK(-21902
    )+256*PEEK(-21901):HTAB12:P
    RINT"Q$"LENGTH = "Q$"PEEK(-
    21920)+256*PEEK(-21919)"
```

Program 4: Memory Map

```
49 REM  MEMORY MAP
50 PRINT "TEXT:HOME"
60 PRINT "PRINT"Q$"MEMORY MAP"Q
   $":PRINT:PRINT"Q$"HIMEM"Q$":
   PRINT"Q$"STRINGS (DOWN TO):"
   Q$":PRINT"Q$"(FREE SPACE)"Q$
   ":PRINT"Q$"ARRAYS, POINTERS"
   Q$": PRINT "Q$"& VARIABLES (
   UP TO):"Q$":PRINT"Q$"LOMEM"Q
   $":PRINT"Q$"PROGRAM LINES TO
   :"Q$
70 PRINT "POKE32,22:VTAB3:PRINT
   :PRINT:PRINTPEEK(115)+256*PE
   EK(116):PRINTPEEK(111)+256*P
   EEK(112):PRINT:PRINT:PRINTPE
   EK(109)+256*PEEK(110):PRINTP
   EEK(105)+256*PEEK(106):PRINT
   PEEK(175)+256*PEEK(176)"
80 PRINT "POKE32,0"
```

©

Atari Password

Glenn Anderson

Would you like to protect a diskful of important programs from prying eyes? If so, here's a solution that discourages all but the most determined snoops: a security program that denies access to your disk unless the correct password is entered. Even if someone boots from a different disk and bypasses the security program, your BASIC listings remain unreadable. For all Atari 400/800, XL, and XE computers with at least 24K RAM, a disk drive, and Atari DOS 2.0, 2.5, or 3.0.

Most people at one time or another have felt the need to protect their programs from prying eyes. At first the solution seems simple: When the program starts, it can ask the user to type in a code which is then compared to a password embedded in the program. If the user fails to enter the right password, the program can end with a NEW command, erasing itself from memory.

This might deflect a rank beginner, but not many other computer users would be fooled. Anyone could obtain the password merely by stopping the program with the BREAK key, typing LIST, and reading through the listing. A password serves no purpose if it can be found so easily.

To keep people from stopping the program and scanning the listing, you can disable the BREAK key by adding this line:

```
1 POKE 16,64:POKE 53774,64
```

Now if the user hits BREAK, nothing happens.

The next thing a persistent person will do, however, is press the Atari's SYSTEM RESET button. The computer does what is called a *warm start*, and the program stops. Since the program is still in memory, the user can type LIST and start looking for the password.

To prevent this from happening, you can add this line:

```
2 POKE 580,1
```

Now when SYSTEM RESET is pressed, the computer does a *cold start* instead of a warm start. It has the same effect as switching the power off and then on again, erasing any program in memory, rebooting the disk operating system (DOS), and loading and running an AUTORUN.SYS file if one is present on the DOS disk. With BREAK and SYSTEM RESET now safely disabled or trapped, the user can't stop your program and discover the secret password.

It's Still Vulnerable

But that assumes your program is running. The user can simply load the program without running it, then type LIST. To prevent this, you could make the BASIC program run automatically on powerup by writing a machine language booter or creating an AUTORUN.SYS file with the autobooting utility included with DOS 2.5. Whenever the computer is booted with this disk, the program automatically runs, and the user must enter the correct password to gain access to the rest of the program.

This works if the user boots with that disk. But what's to stop people from booting with another disk? They can easily gain control of BASIC, insert your disk, load your program, and find the password.

What's really needed is a way to save the program so that it can be run but not loaded. A method for this has already been found and published by COMPUTE! Books in *Mapping the Atari*, and similar solutions have appeared in other publications. To protect a program from being loaded, these two lines must be added:

```
32766 FOR VARI=PEEK(130)+PEEK(131)
      *256 TO PEEK(132)+PEEK
      (133)*256:POKE VARI,155:NEXT
      VARI
32767 POKE PEEK(138)+PEEK
      (139)*256+2,0:SAVE
      "D:filename.ext":NEW
```

It is important to make these the last two lines in the program. The first line fills the variable name table with RETURN characters. The second line finds the location in memory of the current statement line—line 32767 in this example—and POKEs the value of zero into the length of that line. Now, when the computer tries to access a statement with a line number higher than 32767, it gets caught at line 32767 when searching for the line.

This keeps the program from being loaded because of the way Atari BASIC handles an immediate mode command—it treats the immediate mode line as if it were numbered 32768. Since 32768 is higher than 32767, the computer never finds the immediate mode line and never executes it. Therefore, unless the computer is executing the program, the system is effectively crashed because nothing can be done in immediate mode. With this done, the only way to get the program into memory without crashing the system is to run the program at the same time it is loaded from disk: RUN "D:*filename.ext*".

To lock and save a program in this manner, you enter GOTO 32766. The routine saves the program with the filename you specified in line 32767. It also erases the program from memory with NEW, so it's a good idea to save an unprotected copy on another disk before protecting it in case the program needs revisions or debugging.

The Keeper Of The Keys

Now we've got the basis of a password program that can be used to keep out unwanted users. And, thanks to the AUTORUN.SYS loader, the program runs automatically when the disk is booted.

Another idea is to make this autoboot program a menu program that can run other protected programs. This saves the trouble of adding a password procedure to all the protected programs on the disk. To let the other programs know that the user has successfully entered the correct password, the menu program can POKE some arbitrary but predetermined number into any location in an area of memory that is not erased when a new program runs. Then the first line of the new program can check this location for the proper value. If the location does not hold the correct value, the program can stop with a NEW command or rerun the menu program.

You might also want to make the other programs rerun the menu program when they're finished. If this is done, it's wise to have the menu program check the secret memory location for the desired value again so that it knows whether the password has been successfully entered already. If it has, the menu program can skip over the password procedure.

"Atari Password," listed below as Program 1, does all this and a little more. It also includes a way to change the password and unprotect the program.

When typing Program 1, be especially careful with the DATA statements in lines 1–6. They contain information for restoring the variable name table when unprotecting the program.

The initial password is in line 120: ENTER. Type this line exactly as it appears. If you want to change the password later, do it with the option provided for this purpose when running Atari Password, *not* by changing line 120.

Creating A Password Disk

When you've finished typing in Program 1, follow these steps *before running the program*:

1. LIST at least one copy of Atari Password on a backup disk with the command LIST "D:*filename.ext*". Retain this copy as your unprotected backup. Use any filename you like except AUTORUN .BAS, because that's the name used by the protected version of Atari Password.

2. Don't run the program yet. After saving your backup, type NEW to erase it from memory. Then type in Program 2, "Autoboot Maker," and save at least one copy of that program on your backup disk. Don't run this program yet, either.

3. Type NEW to erase Program 2 from memory. Reload Atari Password (Program 1) from your backup disk with the command ENTER "D:*filename.ext*". This ensures that the variable name table will be in the proper state so that the program can be unprotected properly.

4. Now you're ready to create a protected version of Atari Password. Insert a formatted disk that contains Atari DOS 2.0, 2.5, or 3.0. This will be your protected password disk.

5. Type GOTO 9500 and press RETURN. After a brief pause, Atari Password saves a protected version of itself on the disk with the filename AUTORUN.BAS. When it's done, it erases itself from memory.

6. Remove the password disk and insert the backup disk. Load Program 2.

7. Remove the backup disk and insert the password disk. Run Program 2. It creates an AUTORUN.SYS file on the password disk and informs you when it's done. If you've made a typing mistake in the DATA statements, it notifies you of your error. On power-up, this AUTORUN.SYS file runs a BASIC program named AUTORUN.BAS—the protected version of Atari Password. (Note that if there's already an AUTORUN.SYS file on the disk, it will be replaced by this AUTORUN.SYS. Rename or move the existing AUTORUN.SYS to another disk if you don't want to lose it.)

8. The password disk is now prepared. To confirm that Atari Password is working properly, turn the power off, then on again to boot the disk. Atari Password should automatically load and run. You should be able to gain access to the program by typing the default password, ENTER, and then pressing the START button (do not press RETURN). Type the password carefully; the actual keys you press are not echoed on the screen, so it's easy to make a typing mistake. The SELECT button backspaces, and OPTION erases the entire input line. If you accidentally hit the CAPS key, the program may not recognize your password; uppercase and lowercase are significant. If the program denies access with a LOCKOUT message, press START to try again or SYSTEM RESET to reboot.

Using Atari Password

Once you've gained access, Atari Password presents a short menu. Press 1, 2, or 3 for your choice:

1 EXIT TO BASIC
2 CHANGE PASSWORD CODE
3 DISK DIRECTORY

Option 1 exits Atari Password, erases the program from memory, and leaves you in BASIC.

Option 2 lets you change the password from the default—ENTER—to anything you wish. When using this option, make sure you have the password disk inserted in the drive. It rewrites the part of Atari Password which checks for the code word. You can enter any combination of letters or numbers for the password, but it should be no more than 28 characters long.

Option 3 calls up a disk directory on the screen. From this directory, you can load and run any BASIC program saved on the disk. To pick a program, move the arrow pointer with the cursor keys (you don't have to hold down CTRL as you normally do when moving the cursor in BASIC). Then press RETURN. If Atari Password can't load and run the program for some reason—perhaps it's not a BASIC program, or it's saved in LIST format—you're informed of this and allowed to pick another program. To return to the main menu, press the SELECT button.

Protecting BASIC Programs

To protect an ordinary BASIC program and make it dependent on Atari Password, follow these steps:

1. Type in these three lines and LIST them to disk:

```
0 IF PEEK(1612)<>126 THEN RUN"D:
   AUTORUN.BAS"
32766 FOR VARI=PEEK(130)+PEEK
   (131)*256 TO PEEK(132) +PEEK
   (133)*256:POKE VARI,155:NEXT
   VARI
32767 POKE PEEK(138)+PEEK(139)*
   256+2,0:SAVE"D:filename.ext"
   :NEW
```

(Notice line 0; it checks to see whether memory location 1612 contains the value 126. If not, it reruns Atari Password, a protection technique that we mentioned above. If you change Atari Password to put a different number in this location, or if you change the location, be sure to make the change here also.)

2. Load the BASIC program you want to protect. Be sure it doesn't already contain lines numbered 0, 32766, or 32767. Then merge the above lines with your BASIC program by ENTERing the lines from disk.

3. Change the *filename.ext* in line 32767 to whatever name you wish to use for the protected version of your program.

4. Make sure the password disk is inserted in the drive. Type GOTO 32766 and press RETURN. When the READY prompt reappears, the program is protected. Now it can be run only after the password has been successfully entered with Atari Password.

Some password-protected programs have what's known as a *back door.* This is a secret way to bypass the protection. Atari Password doesn't have a back door, but it does have a secret feature that lets you exit the password program to BASIC without erasing the program from memory. When the main menu is on the screen, press the S key and wait. After five or ten seconds, the buzzer sounds. Then press the B key. You'll find yourself in BASIC with the password program intact.

How It Works

Here's a breakdown of Atari Password:

Lines	
1–6	These are DATA statements to refill the variable name table.
60–110	Initialization. Note the variables TRUE and FALSE which assign values to Boolean variables.
120	Contains the password. See explanation below for lines 6000–6080.
500	Checks to see whether the program has been previously run. If not, it checks for password. The memory address (1612) and code value (126) in this line can be changed to any free location and value that you would like, but be sure to reflect your change in line 5000 also.
510	The program reaches here only if the correct password has been entered. If so, it jumps to the main menu.
1000–1160	The main loop of the password-checking routine.
1200–1220	Backs up one space if SELECT is pressed while a password is being entered.
1300	Erases the entire input line if OPTION is pressed while a password is being entered.
1500–1510	Checks for the correct password if START is pressed.
1600–1650	Sounds alarm and displays the LOCKOUT message until START is pressed if the incorrect password is entered.
1999	Returns from the password-checking routine.
2000–2010	Fills the screen with inverse spaces.
2500–2600	Gets the actual password code from C$ and puts it in CODE$.
3000–3210	Main menu section.
6000–6080	This routine changes the password code. It does this by opening Atari Password on disk for read and write and searching for the occurrence of the two Z's that can be seen in line 120. When it finds this flag, it writes the new password code to disk. Something to note is the #16 in line 6005; this suppresses the question mark which is the normal INPUT prompt.
7000–7460	This routine calls the disk directory. It prints the directory on the screen along with an arrow-shaped pointer that can be moved to the desired filename. RETURN runs the selected program. If the entire directory cannot fit on the screen, the message <MORE> appears. Press START to see the rest of the directory or SELECT to go back to the main menu. This routine can be removed and used in your own programs, but remember to DIMension the variables DIR$, ENT$, and PR$, and use a GRAPHICS 0 statement, because the routine uses LOCATE to read the filename from the screen.
9000–9030	Restores the program to BASIC with the listing intact. This is done by refilling the variable name table with its original values, which are stored as DATA statements. Then it POKEs the correct length into line 9510. For this reason, you shouldn't change any of the variables in this program.
9500–9510	This short routine creates the protected version of the password program on disk.

For instructions on entering these listings, please refer to "COMPUTE!'s Guide to Typing In Programs" in this issue of COMPUTE!.

Program 1: Atari Password

```
FD 1 DATA 84,82,85,197,70,65
     ,76,83,197,83,84,65,82,
     212,83,69,76,69,67,212,
     79
PC 2 DATA 80,84,73,79,206,84
     ,82,89,164,67,79,68,69,
     164,67,164,68,73,82,164
MM 3 DATA 69,78,84,164,80,82
     ,164,208,75,69,89,80,82
     ,69,83,83,69,196,67,79
FP 4 DATA 78,83,79,204,193,2
     01,217,78,67,164,68,79,
```

```
    78,197,68,77,65,216,77,
    79
IG 5 DATA 82,197,68,84,79,20
    8,66,79,212,216,68,69,2
    04,79,217,79,216,214,88
    ,67
JP 6 DATA 200,69,216,88,177,
    83,212,76,201,86,65,82,
    201,202
JN 60 OPEN #1,4,0,"K:"
BJ 70 GRAPHICS 0:POKE 82,2
AA 80 POKE 580,1:POKE 16,64:
    POKE 53774,64
LB 90 SETCOLOR 4,10,0:SETCOL
    OR 2,10,0
PJ 100 TRUE=(1=1):FALSE=(1=0
    ):START=6:SELECT=5:OP
    TION=3
CD 110 DIM TRY$(30),CODE$(30
    ),C$(32),DIR$(1235),E
    NT$(17),PR$(14),NC$(3
    0)
MH 120 C$="ZZENTER**********
    ***************"
GA 500 IF PEEK(1612)<>126 TH
    EN GOSUB 2500:GOSUB 1
    000:POKE 1612,126
JC 510 GOTO 3000
IA 997 END
PN 999 REM GET PASSWORD CODE
LH 1000 POKE 752,1:GOSUB 200
    0:POSITION 2,2:? "CO
    DE:";:POSITION 7,2
FN 1010 P=0:TRY$=""
NP 1040 P=P+1:KEYPRESSED=FAL
    SE
DM 1045 IF P=30 THEN 1300
CL 1050 IF PEEK(764)<>255 AN
    D PEEK(764)<>154 THE
    N KEYPRESSED=TRUE
FC 1060 CONSOL=PEEK(53279)
OJ 1065 IF CONSOL=START THEN
     1500
BE 1070 IF CONSOL=SELECT THE
    N 1200
DD 1075 IF CONSOL=OPTION THE
    N 1300
DG 1100 IF  NOT KEYPRESSED T
    HEN 1050
GE 1110 GET #1,A
FJ 1130 TRY$(P)=CHR$(A)
HP 1140 PRINT "■";
MG 1160 GOTO 1040
DE 1198 REM BACKSPACE
HK 1200 IF P<=1 THEN GOTO 10
    50
EP 1210 P=P-1:TRY$(P)="":? "
    {BELL}{LEFT}■{LEFT}";
ME 1220 GOTO 1050
EH 1298 REM CLEARLINE
EG 1300 POSITION 7,2:FOR I=1
     TO P:? "■";:NEXT I:
    TRY$="":POSITION 7,2
    :P=1:GOTO 1050
DK 1498 REM CHECK ENTERED CO
    DE
FK 1500 PRINT "{ESC}
    {CLR TAB}";:FOR I=1
    TO 200:NEXT I
CM 1505 IF TRY$<>CODE$ THEN
    1600
NM 1510 GOTO 1999
IB 1598 REM NO GOOD
AM 1600 GOSUB 2000:POSITION
    16,12:? "LOCKOUT"
HF 1610 SOUND 1,50,10,10:POK
    E 755,2:FOR I=1 TO 5
    0
JH 1620 IF PEEK(53279)<>STAR
    T THEN NEXT I
KB 1630 SOUND 1,100,10,10:PO
    KE 755,0:FOR I=1 TO
    50
NE 1640 IF PEEK(53279)<>STAR
    T THEN NEXT I:GOTO 1
    610
FP 1650 POP :SOUND 1,0,0,0:P
    OKE 755,2:GOTO 1000
LM 1999 RETURN
HH 2000 ? "{CLEAR}";:FOR Y=1
     TO 23:? "
    {38 SPACES}";:NEXT Y
KD 2010 RETURN
IH 2500 P=3
HN 2510 IF C$(P,P)="*" THEN
    2600
MJ 2520 CODE$(P-2)=C$(P,P)
AD 2530 P=P+1
MM 2540 GOTO 2510
KI 2600 RETURN
PF 2998 REM MENU
GA 3000 PRINT "{CLEAR}{38 N}
    ";
MK 3010 PRINT "{18 SPACES}ME
    NU"
LP 3020 PRINT "{38 M}";
AI 3040 PRINT " 1 EXIT TO BA
    SIC"
PP 3050 PRINT " 2 CHANGE PAS
    SWORD CODE"
KN 3060 PRINT " 3 DISK DIREC
    TORY"
OK 3070 PRINT "{38 N}";
GM 3080 GET #1,A
BA 3090 IF A=ASC("S") THEN G
    OSUB 9000
LN 3100 IF A=ASC("B") AND LI
    =-999 THEN GRAPHICS
    0:END
LO 3200 IF A<49 OR A>51 THEN
     3080
CA 3210 ON A-48 GOTO 5000,60
    00,7000
FL 5000 GRAPHICS 0:POKE 580,
    0:POKE 1612,0:NEW
MB 5999 REM CHANGE CODE WORD
OP 6000 POKE 752,0
NF 6005 ? "{CLEAR}ENTER NEW
    CODE PASSWORD(S)";:I
    NPUT #16,NC$
FJ 6010 OPEN #2,12,0,"D:AUTO
    RUN.BAS"
AB 6020 IF LEN(NC$)>29 THEN
    NC$(29)=""
GL 6030 FOR I=LEN(NC$)+1 TO
    30:NC$(LEN(NC$)+1)="
    *":NEXT I
JI 6040 GET #2,A:IF A=90 THE
    N GET #2,A:IF A=90 T
    HEN 6060
MO 6050 GOTO 6040
NC 6060 FOR I=1 TO 30:PUT #2
    ,ASC(NC$(I,I)):NEXT
    I
JI 6070 CLOSE #2
PM 6075 POKE 752,1
MK 6080 GOTO 3000
KA 6999 REM DIRECTORY ROUTIN
    E
HP 7000 OPEN #5,6,0,"D:*.*"
KI 7010 DIR$(1)=" ":DIR$(123
    5)=" ":DIR$(2)=DIR$
JH 7020 DONE=FALSE
PE 7030 POKE 752,1
II 7040 P=0
AF 7050 P=P+1
OM 7060 INPUT #5,ENT$
MH 7070 IF ENT$(5,8)="FREE"
    THEN DONE=TRUE
HM 7080 IF DONE THEN 7110
PG 7090 DIR$(P*19-18,P*19)=E
    NT$
MN 7100 GOTO 7050
II 7110 CLOSE #5:DMAX=P
BB 7120 P=1:MORE=FALSE:IF DM
    AX>40 THEN MORE=TRUE
FN 7130 DTOP=40:IF  NOT MORE
     THEN DTOP=DMAX
GM 7140 POKE 82,2:POKE 83,39
    :PRINT "{CLEAR}
    {13 SPACES}DIRECTORY
    {15 SPACES}"
JL 7150 PRINT DIR$(P,DTOP*19
    );:IF DTOP/2<>INT(DT
    OP/2) THEN PRINT
DA 7160 BOT=PEEK(84)-1
KA 7170 ? " {36 R}";:IF MORE
     THEN POSITION 17,22
    :PRINT "<MORE>";
BK 7180 X=1:Y=1:POSITION X,Y
    :? "{ESC}{RIGHT}"
HI 7190 KEYPRESSED=FALSE
DB 7210 IF PEEK(764)<>255 TH
    EN KEYPRESSED=TRUE
FG 7220 CONSOL=PEEK(53279)
MB 7230 IF CONSOL=START AND
    MORE AND P<>1 THEN 7
    120
KC 7240 IF CONSOL=START AND
    MORE THEN P=(DTOP+1)
    *19+1:DTOP=DMAX:GOTO
     7140
BO 7245 IF CONSOL=SELECT THE
    N 3000
FL 7250 IF KEYPRESSED THEN 7
    270
NC 7260 GOTO 7210
IE 7270 OY=Y:OX=X
EN 7280 GET #1,V:IF V=45 THE
    N Y=Y-1:IF Y<1 THEN
    Y=1
ME 7290 IF V=61 THEN Y=Y+1:I
    F Y>BOT THEN Y=BOT
HJ 7300 IF V=43 OR V=42 THEN
     XCH=XCH+1:IF XCH/2=
    INT(XCH/2) THEN X=1:
    GOTO 7320
BE 7310 IF V=43 OR V=42 THEN
     X=20
IH 7320 IF V=155 THEN 7350
PN 7330 POSITION OX,OY:? " "
    :POSITION X,Y:? "
    {ESC}{RIGHT}"
NI 7340 GOTO 7190
PP 7350 EX=0:PR$="D:":FOR X1
    =X+3 TO X+13
ND 7360 LOCATE X1,Y,V:IF V=3
    2 THEN 7400
BP 7370 PR$(LEN(PR$)+1)=CHR$
    (V)
AF 7380 IF X1=X+10 THEN 7400
NK 7390 GOTO 7430
NO 7400 IF EX=1 THEN EX=0:GO
    TO 7420
KL 7410 X1=X+11:LOCATE X1,Y,
    V:IF V<>32 THEN PR$(
    LEN(PR$)+1)=".":X1=X
    1-1:EX=1:GOTO 7430
NF 7420 GOTO 7440
JG 7430 NEXT X1
ML 7440 TRAP 7450:RUN PR$
EM 7450 POSITION 13,23:? "CA
    NNOT BE RUN";:GET #1
    ,A
NE 7460 GOTO 7120
LO 8990 REM
FN 8991 REM THIS ROUTINE IS
    NOT ON THE MENU
AO 8992 REM IT RESTORES THE
    PROGRAM TO BASIC WIT
    H
HP 8993 REM THE LISTING INTA
    CT
OO 8994 REM TO USE THIS OPTI
    ON PRESS S AT THE ME
    NU
```

```
KG 8995 REM AND THEN PRESS B
         WHEN YOU HEAR THE B
        UZZER
MA 9000 ST=PEEK(136)+PEEK(13
        7)*256
BH 9010 LI=PEEK(ST)+PEEK(ST+
        1)*256:IF LI<>9510 T
        HEN ST=ST+PEEK(ST+2)
        :GOTO 9010
MH 9020 IF PEEK(ST+2)=0 THEN
         POKE ST+2,72:RESTOR
        E 1
DI 9030 FOR I=1 TO 115:LI=PE
        EK(130)+PEEK(131)*25
        6+(I-1):READ VARI:PO
        KE LI,VARI:NEXT I:LI
        =-999:? "{BELL}":RET
        URN
DI 9498 REM PROTECT SAVE ROU
        TINE
OG 9500 FOR VARI=PEEK(130)+P
        EEK(131)*256 TO PEEK
        (132)+PEEK(133)*256:
        POKE VARI,155:NEXT V
        ARI
EP 9510 POKE PEEK(138)+PEEK(
        139)*256+2,0:SAVE "D
        :AUTORUN.BAS":NEW
```

Program 2: Autoboot Maker

```
PA 10 OPEN #1,8,0,"D:AUTORUN
      .SYS"
PN 20 TRAP 40
BJ 30 READ A:PUT #1,A:CHK=CH
      K+A:GOTO 30
GJ 40 IF CHK<>10833 THEN ? "
      Error in DATA statemen
      ts!":END
PE 50 ? "AUTORUN.SYS file ha
      s been written."
CF 1000 DATA 255,255,0,6,109
        ,6
IN 1010 DATA 169,5,141,197,2
        ,133
GJ 1020 DATA 84,169,49,141,6
        8,3
BI 1030 DATA 169,6,141,69,3,
        169
CF 1040 DATA 0,141,73,3,169,
        61
IG 1050 DATA 141,72,3,169,11
        ,141
PI 1060 DATA 66,3,162,0,32,8
        6
ME 1070 DATA 228,169,0,133,8
        4,133
GE 1080 DATA 85,169,13,141,7
        4,3
DP 1090 DATA 96,71,82,46,49,
        43
DF 1100 DATA 49,54,58,63,35,
        54
FG 1110 DATA 59,34,32,32,32,
        207
FB 1120 DATA 206,197,160,205
        ,207,205
FN 1130 DATA 197,206,212,174
        ,174,174
ED 1140 DATA 34,58,80,79,75,
        69
CO 1150 DATA 32,56,52,50,44,
        49
DN 1160 DATA 50,58,82,85,78,
        34
EO 1170 DATA 68,58,65,85,84,
        79
EH 1180 DATA 82,85,78,46,66,
        65
OK 1190 DATA 83,34,226,2,227
        ,2,0,6
```

©