Buyer's Guide To Music Software

COMPUTE!

$3.00
May
1987
Issue 84
Vol. 9, No. 5
$4.25 Canada
02193
ISSN 0194-357X

The Leading Magazine Of Home, Educational, And Recreational Computing

Rememory
Put your memory to the test with this challenging one- or two-player game for the Apple II, Amiga, Atari, Commodore 64, and IBM PC/PCjr

Font Printer For The IBM PC/PCjr
A comprehensive package for designing and printing your own custom fonts

Synthesis
Transform your 64 into a powerful, multifeatured musical synthesizer

ShapeMaker For Apple II
Easily create and edit shape tables

Six New Operators For Atari BASIC

Omega Sort For Commodore 64

Atari Disk Sector Editor

Hi-Res Text For Apple II

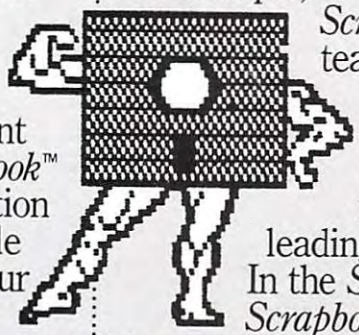
Commodore 128 Colorswap

05
0 71486 02193 3

There are two things almost everyone has in common. An active imagination. And a tough time putting it on paper.

But now we present our *Graphics Scrapbook*™ series. A huge collection of pictures that enable you to easily bring your creative inspirations to The Print Shop,™ PrintMaster™ or Create a Calendar.

On each disk there are over 100 unique pictures. For example, in our *School Scrapbook,* teachers and students will find everything from cheerleading to finals. In the *Sports Scrapbook,* dozens of sports, mascots and trophies. In the *Off the Wall Scrapbook,* well, just expect the unexpected. And plenty more Scrapbooks are coming. So even if you can't draw a straight line, it's okay.



Over a hundred eye-catching pictures on every disk.

As long as you make a straight one to the offer below.

Apple & Compatibles, C64/128, IBM & compatibles.

ALLOW US TO DRAW YOUR ATTENTION.

Time flies. It might as well look snazzy as it goes by.

Introducing *Create A Calendar.*

Our simple, powerful tool that lets you design daily, weekly, monthly or yearly calendars. In moments.

Among numerous features, it includes graphics, borders and fonts for just about every occasion. Or use your own pictures from the Graphics Scrapbooks or Print Shop compatible graphics disks.

What's more, it lets you write multiple lines of text on every date. To make it easy to keep track of everything you're supposed to do. And everything you've done. So this year, go ahead and make your own calendars.

There couldn't be a nicer way to pass the time. EPYX®

USE OUR PREVIEW DISK TO DRAW YOUR OWN CONCLUSION. *Send your name, address, phone number, computer type, and a check or money order for $1.50 to Calendar/Scrapbook Preview, P.O. Box 8020, Redwood City, CA 94063. Canadian orders add 50¢ postage. 6-8 weeks delivery. Expires 8/30/87. Valid only in cont. U.S. and Canada.*

We'll pay you to take the most exciting classes anywhere.

You'll learn electronics, avionics, aircraft maintenance, health care sciences, management or logistics—the Air Force will train you in one of more than 200 technical specialties America needs today.

You'll get hands-on experience with the latest equipment, and we'll pay 75% of your tuition for off-duty college courses, to get you even further.

Whatever your goals, the Air Force will equip you with the skills to get where you want to be.

If you're looking seriously into your future, Aim High to a future in the Air Force. Visit your Air Force recruiter today or call toll-free 1-800-423-USAF (in California 1-800-232-USAF).

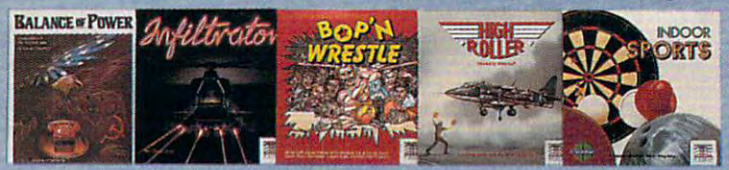
AIR FORCE

STAY HOME AND PLAY.

Why go out when you can have so much fun at home? Just take a gander at the kind of excitement Mindscape has to offer.

With *Indoor Sports*, you can play darts without putting holes in your walls, ice an opponent in air hockey, become a ping-pong pro, and pick up some spares without venturing into an alley.

As a Harrier jump-jet ace in *High Roller*, you'll be doing barrel rolls toward designated targets without waking the neighbors.



Bop'n Wrestle puts you in the ring with 10 of the biggest, baddest bruisers ever to perfect the turnbuckle fly.

Prepare to take evasive action with *Infiltrator*. Foil your foes from your 'copter's cockpit and then convert to covert ground action behind enemy lines. In *Balance of Power*, you are the President. And the burden of global responsibility seems so real you may wonder why you don't have Secret Service protection.

What do you have to lose? For much less than the cost of a night on the town, Mindscape makes home sweet home a more exciting place to be.

Mindscape
Software that challenges the mind.
MINDSCAPE INC

Indoor Sports is available on C64 & C128. *High Roller* is available on C64 & C128 and Atari ST. *Bop'n Wrestle* is available on Apple II family, IBM & compatibles, C64 & C128 and Atari 800. *Infiltrator* is available on Apple II family, IBM & compatibles, C64 & C128 and Atari 800. *Balance of Power* is available on Apple II family, Macintosh, IBM & compatibles, Amiga and Atari ST.

Visit your retailer or call 1-800-443-7982 (in Illinois 1-800-654-3767) for VISA or MasterCard orders. To purchase by mail, send VISA or MasterCard number with expiration date, check or money order to Mindscape, Inc., P.O. Box 1167, Northbrook, IL 60065. Add $3.00 for shipping and handling. Allow 3-5 weeks for delivery.

COMPUTE!

MAY 1987
VOLUME 9
NUMBER 5
ISSUE 84

GUIDE TO ARTICLES AND PROGRAMS

FEATURES

REVIEWS

COLUMNS AND DEPARTMENTS

THE JOURNAL

NOTE: See page 122 before typing in programs.

AP Apple, **Mac** Macintosh, **AT** Atari, **ST,** Atari ST, **64** Commodore 64, **128** Commodore 128, **P** PET/CBM, **PC** IBM PC, **PCjr** IBM PCjr, **AM** Amiga. *General interest.

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

COMPUTE! Publications, Inc.
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

Apple has announced two new Macintosh computers. Their impressive specifications will further strengthen the already impressive Macintosh line: More than one million Macs have been sold and are currently selling at the formidable rate of over 50,000 units a month. What's more, these machines establish new performance standards which foreshadow the consumer computer of tomorrow.

The older machines, the Macintosh 512e and Macintosh Plus, should continue to sell well to home, educational, and business buyers, their traditional markets. The new machines are expected to open new markets for Apple: advanced graphics workstations, memory- or speed-intensive business applications, scientific research, artifical intelligence studies, and other applications not ordinarily associated with "personal" computers. In fact, these new computers diverge in several ways from the traditional Macintosh line as well as from the traditions of home and personal computing.

The Macintosh SE (for System Expansion) is the long-awaited, open-architecture Mac which allows the attachment of third-party peripherals through one expansion slot. The SE also permits the addition of a variety of keyboards because it includes the Apple II-style interface. Although quite similar to the Macintosh Plus, the SE features greater speeds with some software, permits add-ons, and Apple expects it to compete effectively against the PC AT and AT clones. Two important improvements over the Mac Plus derive from adjustments to the ROM routines and system software as well as a significant increase in hard disk communications.

The Macintosh II is higher-end and is targeted to compete with 80386-class machines and the DEC VAX. It diverges from the Macintosh line in several important respects. Featuring an optional color display with as many as 256 simultaneous colors, this machine makes extraordinary graphics possible since it has a total of 16.8 million different colors available.

The Macintosh II is not an integrated package: The computer itself is in a box similar to the IBM PC's; the video is separate. There are six internal expansion slots. The computer can address more than four gigabytes of memory (limited to two gigabytes of internal RAM). The high-capacity, full 32-bit 68020 processor operates at 16 MHz, twice as fast as the Macintosh Plus.

To further beef up the power of the Mac II, Apple offers a 68881 math coprocessor chip which can improve the speed of floating-point calculations as much as 200 times. Also, the data-transfer rate has been increased to over one million bytes per second.

COMPUTE! columnist and longtime Apple-watcher David Thornburg thinks the Macintosh II represents, in effect, a first step in an entirely new direction for Apple. "Rather than look at the personal computer market and move upward, it seems that Apple looked at the serious workstation market (populated by companies like Sun, Symbolics, Apollo, and others) and brought high performance within the price range of small businesses and university research labs."

The Mac II, Thornburg says, would be quite a bargain for, to take one example, researchers working in artificial intelligence. "For well under $10,000, one can get the Macintosh II with a 40-megabyte hard disk, lots of RAM, and a splendid version of LISP—all this would compete quite handily with systems costing five times as much."

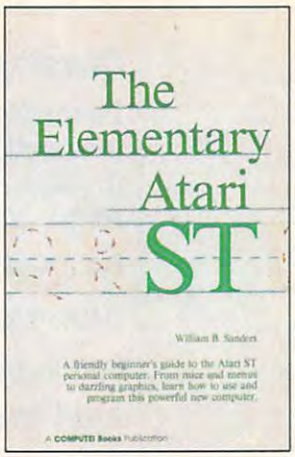
What makes these developments intriguing and even predictive for personal computer users is that we've been here before. Recall the LISA. It was priced beyond most home and educational computer users' budgets and marketed to a similar list of high-end users. But soon after LISA's introduction, the personal computer market was treated to the original Macintosh, with the major design improvements of the LISA intact.

With Apple's announcement of these advanced Macintoshes, and the other high-performance machines coming out of Commodore, Atari, and the IBM world, we can confidently expect to see consumer computers in the next few years which will challenge the capabilities of minicomputers. And all these avant-garde machines seem to have a commonality of design and features, as if the trends of the past several years were now converging and leading to the ideal home computer: extraordinarily impressive graphics resolution, high-quality color, massive memory, open architecture, sophisticated sound capability, and ultra-high speed.

Of course we can always look even farther down the road; no one would mind seeing a consumer version of the massive, state-of-the-art Cray mainframe on a chip. But for the forseeable future, who will be dissatisfied with machines which match the capabilities of all but the most sophisticated commercial graphics workstations?

Richard Mansfield
Editorial Director

Outstanding Artistic Instructive books from COMPUTE!



You'll find expert information, useful applications, intriguing games, graphics, colorful art, music, programming guides, and more in these new Atari ST-specific books. Beginning to advanced ST users will benefit from the applications and tutorials in each book. And as always, the books are written in COMPUTE!'s clear, understandable style.

COMPUTE!'s ST Applications
Brian Flynn and John J. Flynn
$16.95 ISBN 0-87455-067-X
An excellent assortment of games and applications for business and home, written in BASIC, *COMPUTE!'s ST Applications* is an instant library of programs that every ST owner will want to have. All programs have been fully tested and are ready to type in and use on the Atari 520 or 1040 ST. *There is also an optional disk available for $15.95 which includes the programs in the book.*

COMPUTE!'s ST Artist
Selby Bateman and Lee Noel, Jr.
$18.95 ISBN 0-87455-070-X
A step-by-step guide to creating dazzling graphics and art on the Atari ST personal computer. Using *NEOchrome* and *DEGAS**, this book shows you how to get the most out of these excellent painting and drawing programs. Tips and techniques provide you with the most efficient ways of creating graphics and demonstrate how to produce colorful art. Examples illustrate each step and show off all the visual power of the Atari ST and its graphics software. Information is included on the newest versions of *NEOchrome* and *DEGAS Elite. There is an optional companion disk available for $15.95 which includes artwork from the book.*

COMPUTE!'s ST Applications Guide: Programming in C
Simon Field, Kathleen Mandis, and Dave Myers
$19.95 ISBN 0-87455-078-5
COMPUTE!'s ST Applications Guide: Programming in C is your complete tutorial to designing and writing effective ST application programs. Practical examples show you how to use GEM routines to develop professional-looking applications of your own. Explore topics such as disk files, menus, icons, the mouse, sliders, dialog boxes, programming desk accessories, music, and much more. For intermediate to advanced C programmers.

The Elementary Atari ST
William B. Sanders
$18.95 ISBN 0-87455-024-6
A clear, easy-to-use guide to the Atari ST, this book takes you through everything from connecting your computer, loading programs, and creating graphics and music, to writing your own programs.

* A product of Batteries Included.

Order your Atari ST book today. 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.

NC residents add 5 percent sales tax and NY residents add 8.25 percent sales tax.
Shipping and handling: $2.00 U.S. and surface mail; $5.00 airmail.
Please allow 4–6 weeks for delivery.

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies

COMPUTE! books are available outside the United States from subsidiaries of McGraw-Hill International Book Company.

Publisher	James A. Casella
Founder/Editorial Consultant	Robert C. Lock
Editorial Director	Richard Mansfield
Managing Editor	Kathleen Martinek
Associate Publisher	Selby Bateman
Editor, COMPUTE! & COMPUTE!'s GAZETTE	Lance Elko
Assistant Editor, COMPUTE!	Philip I. Nelson
Production Director	Tony Roberts
Editor, COMPUTE!'s Atari ST Disk & Magazine	Tom R. Halfhill
Technical Editor	Ottis R. Cowper
Assistant Technical Editors	George Miller, Dale McBane
Assistant Editor, COMPUTE!'s Atari ST Disk & Magazine	Todd Heimarck
Assistant Editor, COMPUTE!'s GAZETTE	Rhett Anderson
Assistant Features Editor	Kathy Yakal
Programming Supervisor	Patrick Parrish
Editorial Programmers	Tim Victor, Tim Midkiff, William Chin
Copy Editors	Tammie Taylor, Karen Uhlendorf, Karen Siepak
Editorial Assistant	Caroline Hanlon
Submissions Reviewer	David Hensley
Programming Assistants	David Florance, Troy Tucker
Executive Assistant	Debi Nash
Administrative Assistants	Julia Fleming, Iris Brooks, Mary Hunt, Sybil Agee
Associate Editors	Jim Butterfield, Toronto, Canada Fred D'Ignazio, Birmingham, AL David Thornburg, Los Altos, CA
Contributing Editor	Bill Wilkinson
COMPUTE!'s Book Division Editor	Stephen Levy
Assistant Editor	Gregg Keizer
Director of National Sales	Joseph W. Hatcher
Production Manager	Irma Swain
Art & Design Director	Janice R. Fary
Assistant Editor, Art & Design	Lee Noel
Mechanical Art Supervisor	De Potter
Artists	Robin Case, Kim Potts
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
Customer Service Manager	Diane Longo
Dealer Sales Supervisor	Jose Cruz
Individual Order Supervisor	Cassandra Green
Receptionist	Anita Armfield
Warehouse Manager	John Williams

James A. Casella, President
Richard Mansfield, Vice President, Editorial Director
Richard J. Marino, Vice President, Advertising Sales
Christopher M. Savine, Vice President, Finance & Planning

1987 Editorial Board
Richard Mansfield
Kathleen Martinek
Selby Bateman
Lance Elko
Tom R. Halfhill
Stephen Levy
Robert Lock, Founder and Editorial Consultant

COMPUTE! Publications, Inc. publishes:
COMPUTE!
COMPUTE!'s Gazette
COMPUTE!'s Gazette Disk
COMPUTE!'s Apple Applications Special
COMPUTE!'s Atari ST Disk & Magazine

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

Laser Chess: An exciting, innovative game for the Apple II, Commodore 64, IBM PC, Atari, and Amiga

64 RAMdisk

Atari NoDOS

Full Screen Editor for Apple II

RAM Test for Commodore 128

Fast Fractal Landscapes for IBM PC

Using Amiga Disk-Based Fonts

Subscription Orders & Inquiries

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

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

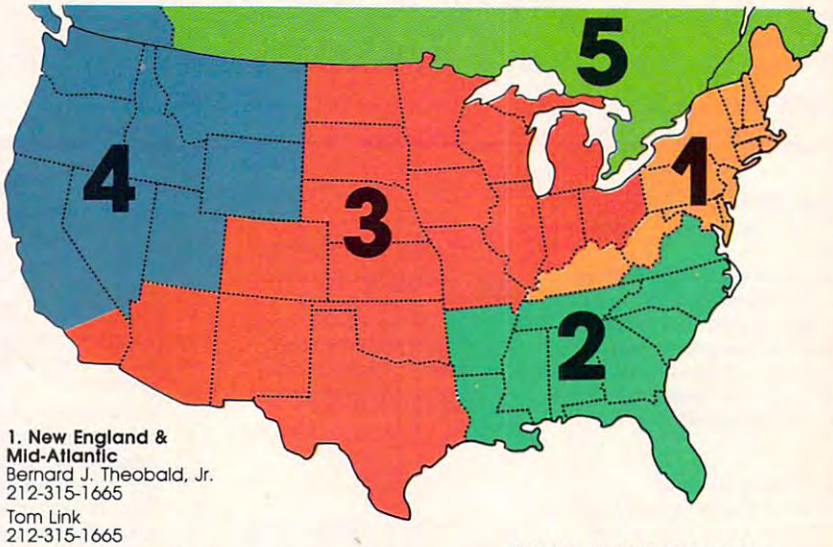
COMPUTE! Subscription Rates (12 Issue Year):

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

Member ABC Audit Bureau of Circulations

MPA Magazine Publishers Association

Advertising Sales



1. New England & Mid-Atlantic
Bernard J. Theobald, Jr.
212-315-1665
Tom Link
212-315-1665

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

3. Midwest & Southwest
Jerry Thompson
312-726-6047 (Chicago)
713-731-2605 (Texas)
303-595-9299 (Colorado)
415-348-8222 (California)
Lucille Dennis
415-348-8222

4. West, Northwest & British Columbia
Jerry Thompson
415-348-8222
Lucille Dennis
415-348-8222

5. Canada
Harry Blair
919-275-9809

Director of Advertising Sales:
Peter Johnsmeyer

Associate Advertising Director:
Bernard J. Theobald, Jr.

COMPUTE! Sales Office 212-315-1665

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

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.

Proofreader For Tandy 1000

I recently purchased a Tandy 1000 IBM-compatible computer and subscribed to your magazine. After numerous attempts, I have been unable to make the "IBM PC/PCjr Proofreader" program work on my computer. I would like to know if there is something in this program that keeps it from working on my computer and, if so, will I have the same problem if I try to enter other IBM programs from your magazine?

Billy Bolden

At COMPUTE! we make every effort to insure that our IBM PC/PCjr programs will also work on the dozens of IBM-compatible models now available. Since we don't have one of each different model available for testing purposes, we can't guarantee that every program will work on every model. Nevertheless, we have found few documented cases where our programs wouldn't work because of machine incompatibility. Most problems occur on systems which lack some required hardware. For example, programs which require a color/graphics adapter card (or equivalent hardware) will not work on a system that has only a monochrome adapter card.

The "Proofreader" is, for the most part, a "plain vanilla" BASIC program, using no fancy programming tricks. The only exception is in line 160, which uses the dynamic-keyboard technique to insure that LINE INPUT gets the entire program line. While it's possible that your computer is incompatible with this program, a much more likely source of your problem is a typing error.

Check every line of the program carefully against the magazine listing, even those lines you believe you typed correctly. Even experienced programmers make typing mistakes, and a single typing error can have drastic effects, depending on where it occurs in the program.

You also should make sure that the computer is in Caps Lock mode (so that all letters appear in uppercase) except when the listing shows that you should be typing a lowercase letter. This is important because the Proofreader is sensitive to the case of characters. These three lines, for instance, generate three different checksums when typed in with the Proofreader. The checksums are shown in front of each line.

HN 10 print "hi there"
NN 10 PRINT "HI THERE"
FN 10 PRINT "Hi there"

In COMPUTE! listings for the IBM PC/PCjr and compatibles, BASIC keywords such as PRINT and IF are always in uppercase. Lowercase letters usually don't appear except after REM and DATA statements, or, as shown in the example, as part of a string enclosed in quotation marks.

We don't have access to an original Tandy 1000, but we do have one of the new Tandy 1000EX models. After receiving your letter, we tested the IBM PC/PCjr Proofreader program on our Tandy 1000EX to confirm that it works correctly. The program works the same on a Tandy 1000EX as it does on an IBM PC/PCjr. The Proofreader has also worked on all the other IBM-compatible models we have tested.

Sending Printer Escape Codes In Amiga BASIC

Here is some information that will be useful to any Amiga owner who wishes to use special printer effects (double strike, compressed characters, and so on) from Amiga BASIC. I have had no difficulty printing Amiga screen dumps on my Epson MX-80. However, when I tried to send printer escape codes from Amiga BASIC, they had no effect. This occurred both when I tried to send the control codes with LPRINT and when I used PRINT# to send output to a file I had previously opened to LPT1:, the printer device. In these circumstances, it appears that all printer output is filtered according to the printer selected in Preferences. The solution is to open a printer file using the PAR: device for a parallel printer or the SER: device for a serial printer. If you then use PRINT# to output CHR$(27) followed by the appropriate control codes, your printer will behave as it should.

Charles Heckel

Thank you for the information. Although the Amiga BASIC manual doesn't mention PAR: or SER:, both device names are understood by AmigaDOS, the disk operating system which BASIC uses for input/output operations.

Upgrading To An Apple IIGS

I am very impressed by what I have read about the Apple IIGS. Do you have any information on how to upgrade a IIe to IIGS specifications? I understand that the 65C816 microprocessor is available to individuals at a reasonable price, and I would like to purchase one and put it all together.

Mike Mendoza

When Apple premiered the IIGS, they also announced that IIe models could be upgraded to IIGS level. Although they offered only complete IIGS systems at first, upgrades should become available sometime in 1987 for around $500.

About upgrading it yourself: The 65C816 chip can't simply replace a IIe's 6502 or 65C02 processor. Its pinouts—the signals which are present on each of the chip's leg-like connecting pins—are different enough from earlier models that they aren't interchangeable. Another new chip, the 65C802, is pin-compatible—you just plug it in, and it runs. It has the same new machine language instructions as the 65C816, but like the older chips, it can only directly access 64K of memory. Although we haven't tried doing it, putting a 65C802 in an Apple IIe sounds like an interesting idea. It would be totally compatible with existing Apple II software, but it wouldn't be much like a IIGS.

Many third-party hardware makers offer plug-in cards for the Apple IIe which contain a 65C816, often along with more memory. These accelerator cards can include quite a bit of RAM, far beyond the IIe's 64K or 128K, as well as a high-speed clock for more processing power. Some of these cards can run eight-bit Apple II software faster than a IIGS in emulation mode. For users who only want more speed and storage for IIe applications like

From impossible dungeons and split-second snares, the Bard and his party emerge. The Sceptre, so long forgotten, gleams with power like an exploding sun. Even Phenglei Kai, the ancient archmage, bows his head in awe.

"I smell serpents!" Slipfinger squeals, stealing away like the thief he is. Two archdragons slither out of the ground, their eyes burning with the relentless fury of treasure lost.

Protected behind the flame lizards, beyond the reach of normal weapons, a cackling wizard begins the eerie chants of a death spell. A spell that can finish the Bard and his party.

The time has come to battle-test the magic of the Destiny Wand – and reveal the awesome powers of The Destiny Knight.™

The Best Ever Dungeon Role-Playing Game

- 50% bigger than Bard's Tale.™
- An all-new story line.
- Six cities and a huge overland wilderness to explore.
- Dozens of new spells – 79 spells in all.
- New real-time dungeon puzzles. You have to get through them before the clock stops ticking.
- Summon and name monsters to become a permanent part of your party.
- More strategy in combat encounters – the weapons and spells you choose depend on the enemy's distance.
- A bank and casino.
- A starter-dungeon for building up your low-level characters.
- 6 guilds for easier game saving.
- Optional use of Bard's Tale characters. Bard's Tale experience not required.
- Cluebooks available for both Bard's Tale and Bard's Tale II.™

You get a new class of magic user – the Archmage. With 8 powerful spells like Heal All, Fanskar's Night Lance, and the awesome Mangar's Mallot.

There are over 100 monsters, like this Kner Drone. Many animated. All dangerous.

25 scrolling dungeon levels. All in color. All 3-D. Including 7 different Snares of Death, a new kind of real-time puzzle.

The Bard's Tale II

The Destiny Knight

from

ELECTRONIC ARTS™

Optional use of characters from Ultima III™ or Bard's Tale™. Apple version uses Wizardry™ characters, too!

HOW TO GET IT: Visit your retailer, or call 800-245-4525 (in CA call 800-562-1112) for VISA or Mastercard orders. To buy by mail, send a check, money order, or VISA or Mastercard information to Electronic Arts, P.O. Box 7530, San Mateo, CA 94403. The price is $39.95 for the Commodore 64 version, and $49.95 for the Apple version. Add $5 for shipping and handling ($7 Canadian). Allow 1-4 weeks for delivery. The Bard's Tale II and Electronic Arts are registered trademarks of Electronic Arts. Ultima is a registered trademark of Richard Garriott. Commodore is a trademark of Commodore Electronics Ltd. Wizardry is a trademark of Sir-Tech Software, Inc. For a copy of our complete catalog, send 50¢ and a stamped, self-addressed envelope to Electronic Arts Catalog, 1820 Gateway Drive, San Mateo, CA 94404.

When you want to talk computers....

HOME COMPUTERS.

Atari Computers
520ST Monochrome System.....$499.00
520ST Color System.....749.00
1040ST Color System.....879.00
800XL 64K Computer.....63.99
65XE 64K Computer.....97.99
130XE 132K Computer.....129.00
Atari Peripherals
1020 Color Printer.....29.99
1050 Disk Drive.....129.00
835 300 Baud Modem.....24.99
850 Atari Interface.....109.00
M301 300 Baud Modem.....39.99
XM801 80-Column Printer.....199.00
XM804 ST Printer.....189.00
ICD PR Connection.....59.99
Xetec Graphix (XL, XE).....39.99

Atari 1040 Color System $879
Includes: 1040ST, 1 mb RAM with 3½" drive built-in, 192K ROM with TOS, Basic, Logo, ST language, power supply and color monitor.

Commodore Computers
Commodore-64C 64K Computer189.00
Commodore-64C System w/1802C539.00
Commodore-128 128K Computer259.00
Commodore-128 System.....759.00
Amiga 500 & 2000.....call
Commodore Peripherals
1660 Commodore Modem.....59.99
1670 Commodore Modem.....99.99
1541C Disk Drive.....189.00
1571 Disk Drive.....239.00
1802 Color Monitor.....199.00
1902 Color Monitor.....299.00
Amiga 1010 3½" Ext. Drive.....219.00
Amiga 1020 5¼" Ext. Drive.....189.00
Amiga 1080 RGB Monitor.....269.00
C128 512K Expansion Board.....179.00
PPI Parallel Printer Interface.....34.99
Xetec S/Graphix 8K.....69.99
Micro R&D MW350.....44.99

MS/DOS SYSTEMS.

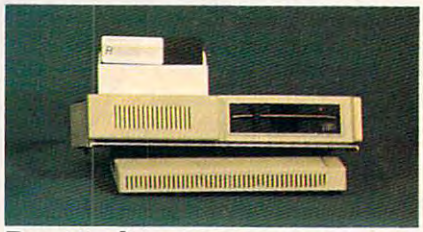
PC-T00 20 Meg XT-Compatible $999
AT&T 6300.....from $1299.00
Compaq.....from 1699.00
Cordata.....from 899.00
IBM-XT.....from 1099.00
IBM-AT.....from 2699.00
Leading Edge.....from 999.00
NEC Multispeed.....from 1499.00
Panasonic Business Partnerfrom 799.00
Toshiba 1100 Plus.....from 1749.00

MULTIFUNCTION CARDS.

AST
Six Pak Plus PC/XT.....$169.00
Six Pak Premium PC/XT.....249.00
Advantage-AT 128K.....339.00
Everex
EV-221 Evergraphics Mono.....139.00
EV-640 Edge Card.....259.00
Hercules
Color Card.....159.00
Graphics Card Plus.....209.00
Fifth Generation
Logical Connection 256K.....329.00
IDEAssociates
IDE-5251 Local Emulator.....579.00
Intel
1110 PC Above Board.....279.00
Inboard 386K 0K.....Call
NEC
GB-1 EGA.....409.00
Quadram
Quad Ega+ Graphics Adapter....299.00
Silver Quadboard.....129.00
Expanded Quadboard.....119.00
VIDEO 7
EGA Video Deluxe.....389.00
Zuckerboard
Color Card w/Parallel.....89.99
Monochrome Card w/Parallel.....99.99
576K Memory Card.....59.99

DRIVES.

Allied Technology
Apple Half-Heights.....$109.00
Controller Card.....39.99
CMS
Drive Plus 20MB Internal Card....399.00
Everex
Stream 20 20MB Tape-Backup....669.00
Genie Technology
210 H 10+10 subsystem.....1749.00
Indus
Atari GT Disk Drive.....189.00
Commodore GT Disk Drive.....189.00
Iomega
A210H 10+10 Bernoulli Box.....1899.00
A220H 20+20 Bernoulli Box.....2499.00
Irwin
110 D 10MB Tape backup.....319.00
Mountain Computer
Drive Card 20MB Internal Card...499.00
A220 20+20 Subsystem.....2199.00



Racore Jr. Expansion Chassis $299

Seagate
ST-225 w/Controller.....399.00
Toshiba
Half-Height 360K internal.....89.99

DISKETTES.

Maxell
MD-1 SS/DD 5¼".....$8.99
MD-2 DS/DD 5¼".....10.99
MD-2HD Hi-Density 5¼".....21.99
MF-1 SS/DD 3½".....12.99
MF-2 DS/DD 3½".....21.49
CS-500 20Mb Streamer Tape.....11.99
CS-600 60Mb Streamer Tape.....13.49
Sony
MD1 SS/DD 5¼".....7.99
MD2 DS/DD 5¼".....9.49
MD-2HD Hi-Density 5¼".....20.49
MFD-1 SS/DD 3½".....12.99
MFD-2 DS/DD 3½".....19.99

COMPUTER MAIL ORDER

.....When you want to talk price.

MONITORS.

Amdek 410 12" TTL Monitor **$159**

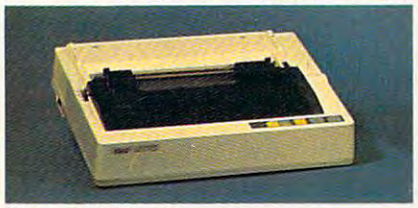
Amdek
Video 310A Amber TTL.....$149.00
Color 722 RGB, CGA/EGA.....479.00
Magnavox
8CM515 RGB Monitor-80.....289.00
7BM623 PC Monitor-80 Amber....99.99
8CM873 14" Multimode.....549.00
Mitsubishi
XC 1409C 14" RGB.....319.00
NEC
12" TTL Green or Amber.....109.00
JC-1401P3A Multi-Sync.....579.00
Princeton Graphics
MAX-12 12" Amber TTL.....169.00
HX-12 12" Color RGB.....429.00
HX-12E 12"RGB/EGA.....499.00
Quadram
8460 Quadchrome Enhanced....439.00
Taxan
Model 124 12" Amber.....119.00

MODEMS.

Anchor
6480 C64/128 1200 Baud.....$119.00
Omega 80 Amiga.....129.00
VM520 ST520/1040 1200 Baud...139.00
Expressi PC-1200 Half Card.....149.00
Everex
Evercom 1200 Baud Internal.....119.00
Hayes
Smartmodem 300 External.....139.00
Smartmodem 1200B Internal.....359.00
Smartmodem 2400B Internal.....539.00
Practical Peripherals
Practical Modem PC-1200.....139.00
Quadram
Quadmodem II 1200 Baud.....299.00
Supra
MPP-1064 AD/AA C64.....69.99
1200AT 1200 Baud Atari.....149.00
U.S. Robotics
2400 Baud Internal.....189.00

PRINTERS.

Canon
LBP-8A1 Laser, 8 Page/Min.....$1899.00
Citizen
MSP-10 160 cps, 80-Column.....319.00
Premier 35 35 cps Daisywheel....499.00
C.Itoh
8510-SP 180 cps, 80-Column.....Call
310-SEP Epson/IBM 80-Column.....Call
Cordata
The Desktop Printshop Laser.....2199.00
Epson
LX-86 120 cps, Dot Matrix.....199.00
FX-86E 240 cps, 80-Column.....Call
FX-286E 240 cps, 132-Column.....Call
EX-800 300 cps, 80-Column.....449.00
LQ-800 180 cps, 24-Wire Printhead..Call
Hewlett Packard
Thinkjet.....399.00
Juki
6300 40 cps Daisywheel.....659.00
6100 10 cps Daisywheel.....389.00
5510C Color Dot Matrix.....349.00
NEC
Pinwriter 660 24 Wire.....489.00
Pinwriter 760 24 Wire.....689.00
Okidata
ML-182 120 cps, 80-Column.....239.00
ML-192+ 200 cps, 80-Column....369.00
ML-193+ 200 cps, 132-Column.....Call
ML-292 200 cps, 80-Column.....Call
Panasonic
KX-1080i 120 cps, 80-Column.....219.00
KX-1091i 180 cps, 80-Column.....299.00
KX-1592 180 cps, 132-Column.....439.00



Star NX-10 120 cps Dot Matrix **$209**

Star Micronics
NX-10C 120 cps, C64 Interface.219.00
NX-15 120 cps, 132-Column.....369.00
Texas Instrument
TI-855 150 cps, 80-Column.....599.00
Toshiba
P321 216 cps, 24-Pin Printhead479.00
P341 216 cps, 24-Pin Printhead589.00

SOFTWARE.

FOR AMIGA
Aegis Development
Animator & Images.....$99.99
Commodore
Textcraft/Graphcraft.....59.99
Electronic Arts
Deluxe Paint.....64.99
Microillusions
Dynamic CAD.....359.00
Micro Systems
Scribble Word Processor.....79.99
Sublogic
Flight Simulator II.....37.99

The Print Shop For IBM **$39.99**
For Commodore & Atari **$29.99**

FOR ATARI ST
Access
Leader Board Golf.....27.99
Batteries Included
D.E.G.A.S. Elite.....59.99
Microprose
Silent Service.....29.99
Paradox
Wanderer 3D.....29.99
Sublogic
Flight Simulator II.....37.99
Timeworks
Swiftcalc.....54.99

FOR IBM
Ashton-Tate
d-Base III+.....429.00
5th Generation
Fastback Utility.....89.99
IMSI
Optimouse w/Dr. Halo.....99.99
Lotus
Lotus 1-2-3.....329.00
MicroPro
Professional 4.0 w/GL.....239.00
Microstuf
Crosstalk XVI.....89.99
P.F.S.
First Choice.....119.00
Satellite Systems
Word Perfect 4.2.....209.00

In the U.S.A. and in Canada

Call toll-free: 1-800-233-8950.

Outside the U.S.A. call 717-327-9575 Telex 5106017898

Educational, Governmental and Corporate Organizations call toll-free 1-800-221-4283

CMO. 477 East Third Street, Dept. A205, Williamsport, PA 17701

ALL MAJOR CREDIT CARDS ACCEPTED.

POLICY: Add 3% (minimum $7.00) shipping and handling. Larger shipments may require additional charges. Personal and company checks require 3 weeks to clear. For faster delivery use your credit card or send cashier's check or bank money order. Pennsylvania residents add 6% sales tax. All prices are U.S.A. prices and are subject to change and all items are subject to availability. Defective software will be replaced with the same item only. Hardware will be replaced or repaired at our discretion within the terms and limits of the manufacturer's warranty. We cannot guarantee compatibility. All sales are final and returned shipments are subject to a restocking fee.

Appleworks, *this option is worth considering.*

But a IIGS has a lot more than a new processor and more RAM. The new video and sound circuitry are only available from Apple. They are also the only source for the ToolBox software built into each IIGS, supporting Macintosh-like windows and menus. Since most commercial developers plan to use all these new features, the only foreseeable way to make your IIe into a true IIGS is through your Apple dealer. And, for this operation, upgrade *isn't exactly the right word. All the IIe's electronic innards are replaced with a new main circuit board. The only parts that are kept are the cabinet, power supply, and keyboard. (Your old interface cards will still work, though.) But when enough new IIGS software has arrived, this procedure could be a very effective means for entering the 16-bit world.*

Quiet Disk Format For Commodore 128

The excellent short program written by Martin Filbeau for the Commodore 64 ("Readers' Feedback," December 1986) does indeed prevent the 1541 disk drive's head from rattling when you format a disk. But that program doesn't work on the Commodore 128 in 128 mode. Here is a modified version of the program that works in 128 mode with either a 40-column or 80-column monitor.

```
AG 5 PRINT"{CLR}{GRN}":COLOR 4
     ,1:COLOR 0,1
MG 10 PRINT"{2 DOWN}INSERT REF
      ERENCE DISK"
QJ 20 GOSUB 270
MP 30 OPEN1,8,15,"IO"
ME 40 OPEN2,8,2,"#"
KR 50 PRINT#1,"U1";2;0;1;0
AQ 60 INPUT#1,N,M$,T,S:PRINT N
      ,M$,T,S
CR 70 IF N=0 THEN 130
GH 80 PRINT N,M$;T;S
GF 90 PRINT"{2 DOWN}TRY AGAIN?
      (Y/N)"
DK 100 GOSUB 280
PF 110 GET F$:IF F$="Y" THEN 5
      0
BG 120 CLOSE 2:CLOSE 1:END
KC 130 PRINT"{4 DOWN}REMOVE RE
      FERENCE DISK"
FB 140 PRINT"{DOWN}INSERT BLAN
      K DISK"
FM 150 GOSUB 270
FM 160 FOR I=1 TO 25
BK 170 READ D:D$=D$+CHR$(D)
MR 180 NEXT
KE 190 PRINT#1,"M-W";CHR$(0);C
      HR$(5);CHR$(25);D$
KA 200 PRINT#1,"M-W";CHR$(32);
      CHR$(6);CHR$(3);CHR$(10
      );CHR$(64);CHR$(15)
PS 210 POKE 208,0
JK 215 PRINT"{CLR}{2 DOWN}"
HR 220 INPUT"DISK NAME";DNAM$
RA 230 INPUT"{2 SPACES}DISK ID
      ";DID$
KJ 240 PRINT"{4 DOWN}FORMATTIN
      G..."
DD 250 PRINT#1,"U3:";DNAM$","D
      ID$
MC 260 GOTO 120
QD 270 PRINT"{4 DOWN}PRESS ANY
       KEY TO CONTINUE"
HK 280 PRINT
JE 290 POKE 208,0
AJ 300 WAIT 208,1
PK 310 RETURN
EC 320 DATA 169,78,141,0,2,169
      ,48,141,1,2,169,11,141,
      42,2
CD 330 DATA 32,238,193,169,1,1
      33,81,76,13,238
```

Carlos Vidales

Thanks for the modification. Because of the length of this program, we've added checksums for our "Automatic Proofreader" program. If you're unfamiliar with the Proofreader, see "COMPUTE!'s Guide to Typing In Programs" elsewhere in this issue.

DOS 3.3 CATALOG From Applesoft

I am using an Apple IIc and would like to know how to read a DOS 3.3 catalog into a BASIC array. Can you show me how to do it and explain how it works?

Steven Pinckney

DOS 3.3, unlike ProDOS, provides no easy way to do this. However, it can be done. The following code adapts part of "Jacket Lister," a program that appeared in the September 1986 issue of COMPUTE!:

```
20 DIM TB$(144),WS(1000)
80 FOR I = 768 TO 779: READ A
   : POKE I,A: NEXT
90 C = 0:P1 = WS(0) - WS(0) +
    PEEK (131):P2 = WS(0) - W
   S(0) + PEEK (132)
100 POKE 769,P1: POKE 770,P2
110 POKE 54,0: POKE 55,3: POK
    E 56,11: POKE 57,3: CALL
    1002
120 PRINT CHR$ (4);"CATALOG"
125 PRINT
130 POKE 768,173: POKE 769,P1
    : POKE 770,P2
140 POKE 54,11: POKE 55,3: PO
    KE 56,0: POKE 57,3: CALL
    1002
150 FOR I = 0 TO 4: INPUT A$:
     NEXT
160 INPUT TB$(C): IF TB$(C) =
    "" THEN 170
165 C = C + 1: GOTO 160
170 POKE 54,240: POKE 55,253:
     POKE 56,27: POKE 57,253:
     CALL 1002
190 DATA 141,0,64,238,1,3,208
    ,3
200 DATA 238,2,3,96
```

The program starts by dimensioning two arrays, TB$ and WS. TB$ is a table of strings to hold the directory entries. WS is just a big block of storage to be used as workspace. Line 80 sets up a short machine language routine which will be used for trapping input and output. The next line uses a trick to find out the address where an array variable is stored. Most versions of BASIC have a function called VARPTR to do this, but Applesoft *doesn't. P1 gets the eight low bits of the address, and P2 gets the eight high bits.*

In line 100, the address of the workspace is stored in the machine language routine. Line 110 hooks up this routine to intercept all input and output operations, then tells DOS about the new I/O routines with the CALL 1002 statement. (Otherwise, DOS would be completely disconnected.) As it's hooked up at this point, the ML routine will store all output in the workspace and disregard requests for input.

Lines 120 and 125 perform the CATALOG operation, printing all the information into the workspace. In the next line, the ML routine is modified slightly to function as an input routine, and the workspace pointer is reset to the start of the WS array. Then the I/O hooks are changed so that input operations will read from the workspace, while output requests will be ignored.

After line 150 skips four header lines, lines 160 and 165 read each catalog entry into the TB$ array. Variable C keeps count of the number of files found. Finally, line 170 resets the I/O hooks to the normal values for a 40-column display, and the program ends.

Phantom Opcodes On The 6502

I have a question about 6502 assembly language. I know that every machine language command is contained in one byte, which may be followed by one or two additional bytes. For example, the byte value for the LDA immediate instruction is 169 ($A9). Some of the possible byte values, however, are not assigned to an instruction. A machine language monitor prints **???** when you try to disassemble one of these instructions. What do these instructions do when the computer executes them? I have heard that they give the combined effect of two other instructions.

Gergely Viczian

Not all of the 256 possible one-byte values are defined as valid machine language instructions for the 6502/6510/8502 microprocessor. The remaining values are officially undefined, meaning that the designers of the processor do not intend them to be used as instructions at all. Many machine language monitors flag such values with ??? to indicate that the byte value could not be interpreted as a valid opcode.

If you've been trying to learn machine language by disassembling other people's programs, you may see many places where it appears on the surface that an undefined opcode has been used.

IS GETTING THE ANSWER TO SOFTWARE PROBLEMS A BIGGER PROBLEM THAN THE PROBLEM?

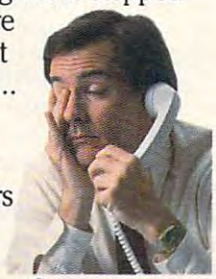
Don't stay on hold when there's help online from CompuServe® Software Forums.

The new upgraded version of your software locks up. And every time you reboot, you get stuck in the same place in the program.

You've chucked the manual, because you've done exactly what it tells you to do six times already. So you call the software company.

Now you spend half a day beating your head against a brick wall of busy signals, ranting at recorded messages, hanging around on hold. And you still don't get the solution to your problem.

Meanwhile, progress is stopped and your profits are dribbling away. But wait. There's help...



Several prominent, progressive software publishers recognize this problem, and working with CompuServe, have developed a solution—CompuServe Software Forums.

Now you can go online with experts from the companies that produced your software and get prompt, written answers to your specific problems. You can even talk with the actual software developers.

Aldus,® Ashton-Tate,® Autodesk,® Borland International,® Creative Solutions,® Digital Research,® Living Videotext,® Lotus® Inc., Microsoft,® MicroPro,® Misosys Inc.® and Software Publishing® all have CompuServe Software Forums. And we keep adding more.

CompuServe's large subscriber base also puts you in touch with thousands of other, often more experienced, users of the same software. You'll find they can give you lots of creative ways to get the most out of your software.

And software forums are the best way to learn about product updates, new product announcements, new ways to expand the uses of your software, and offer free uploads of your own programs.

Our online electronic magazines frequently publish software reviews. And you can find help for many other software products in our other computer-related forums for IBM,® Tandy,® Atari,® Apple,® Commodore,® TI® and others.

The last thing you need when you've got a software problem is a bigger problem getting answers. So, from now on, get prompt, informed answers on CompuServe Software Forums.

To buy your CompuServe Subscription Kit, see your nearest computer dealer. Suggested retail price is $39.95.

To order direct or for more information, call 800-848-8199 (in Ohio, 614-457-0802).

If you're already a CompuServe subscriber, just type GO SOFTWARE at any ! prompt.

CompuServe®

Information Services, P.O. Box 20212
5000 Arlington Centre Blvd., Columbus, OH 43220

An H&R Block Company

Instruction	Abs	Abs,X	Abs,Y	Zer	Zer,X	Zer,Y	(Ind,X)	(Ind),Y	Imm
ASO (ASL,ORA)	0F	1F	1B	07	17		03	13	0B
RLA (ROL,AND)	2F	3F	3B	27	37		23	33	2B
LSE (LSR,EOR)	4F	5F	5B	47	57		43	53	
RRA (ROR,ADC)	6F	7F	7B	67	77		63	73	
AXS (STX,STA)	8F			87		97	83		
LAX (LDX,LDA)	AF		BF	A7	B7		A3	B3	
DCM (DEC,CMP)	CF	DF	DB	C7	D7		C3	D3	
INS (INC,SBC)	EF	FF	FB	E7	F7		E3	F3	
ALR (LSR,EOR)									4B
ARR (ROR,ADC)									7B
OAL (TAX,LDA)									AB
SAX (DEX,CMP)									CB
NOP	1A, 3A, 5A, 7A, DA, FA								
SKB	80, 82, C2, E2, 04, 14, 34, 44, 54, 64, 74, D4, F4								
SKW	0C, 1C, 3C, 5C, 7C, DC, FC								

ASO ASL then ORA the result with the accumulator
RLA ROL then AND the result with the accumulator
LSE LSR then EOR the result with the accumulator
RRA ROR then ADC the result from the accumulator
AXS Store the result of A AND X
LAX LDA and LDX with the same data
DCM DEC memory and CMP the result with the accumulator
INS INC memory then SBC the result with the accumulator
ALR AND the accumulator with data and LSR the result
ARR AND the accumulator with data and ROR the result
OAL ORA the accumulator with #$EE, AND the result with data, then TAX
SAX SBC data from A AND X and store result in X
NOP No operation
SKB Skip byte (that is, branch of +1)
SKW Skip word of two bytes (that is, branch of +2)

However, you should be aware that—in the vast majority of cases—when you see ??? in a section of disassembled code, you are not *looking at a undefined opcode. It's much more likely that you've tried to disassemble a section of memory that doesn't contain machine language, but rather contains data tables, message text, jump vectors, or the like. Since it's only coincidental that the values of these types of data will fall in the range of valid opcodes, most bytes in such areas will show as ???. Undefined opcodes are very rarely used. If you disassemble the entire 16K of BASIC and Kernal ROM in a Commodore 64, you'll find many places where the data disassembles as ???, but* none *of these is truly an undefined opcode.*

Some of the undefined opcodes—sometimes called quasi-opcodes—simply lock up the computer. The computer locks up completely when you attempt to execute any byte ending with $3, $7, $B, or $F, and most byte values ending with $2.

Other undefined opcodes cause the processor to perform a meaningful task. Some of these simply replicate a standard instruction: For example, there are six byte values ($1A, $3A, $5A, $7A, $DA, and $FA) that duplicate the NOP (No OPeration) opcode. Others, such as SKB (SKip a Byte) and SKW (SKip a Word) do jobs that are not done by any standard instruction. The remaining quasi-opcodes generally combine the effects of two standard instructions. For example, the quasi-op LAX loads both the A and X registers with the same value, just as if you had performed LDA and LDX in sequence with the same value.

Quasi-opcodes have few practical uses. You might save a byte here or there by performing two jobs with one instruction, but most quasi-ops perform pretty obscure functions, and since ordinary monitors and assemblers don't allow for them, it's difficult to write or even disassemble programs containing such codes. Because quasi-ops show up as ??? in an ordinary monitor, they have been used occasionally as a concealment device in copy-protected commercial programs. But for ordinary programming, they are probably more trouble than they're worth. The table above lists all the usable quasi-opcodes, taken from Programming the Commodore 64, *by Raeto West (COMPUTE! Books). The codes shown in boldface type are thought to be the most reliable.*

De Re Atari Lives

I just read a letter in the February 1987 installment of this column referring to the availability of the Atari reference book *De Re Atari*. Your readers may be interested to know that a large supply of these books is available for $10 per copy from this computer dealer.

B & C Computervisions
3283 Kifer Rd.
Santa Clara, CA 95051

They also stock many other hard-to-find Atari publications and products.

M. J. White

Thank you for this information.

BUMPing In BASIC 7.0

I am writing a game for my Commodore 128 and I have run across a problem with the BUMP(2) function. From what I can determine, the BUMP values to signal sprite collisions should be as follows:

Sprite	Bump Value
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

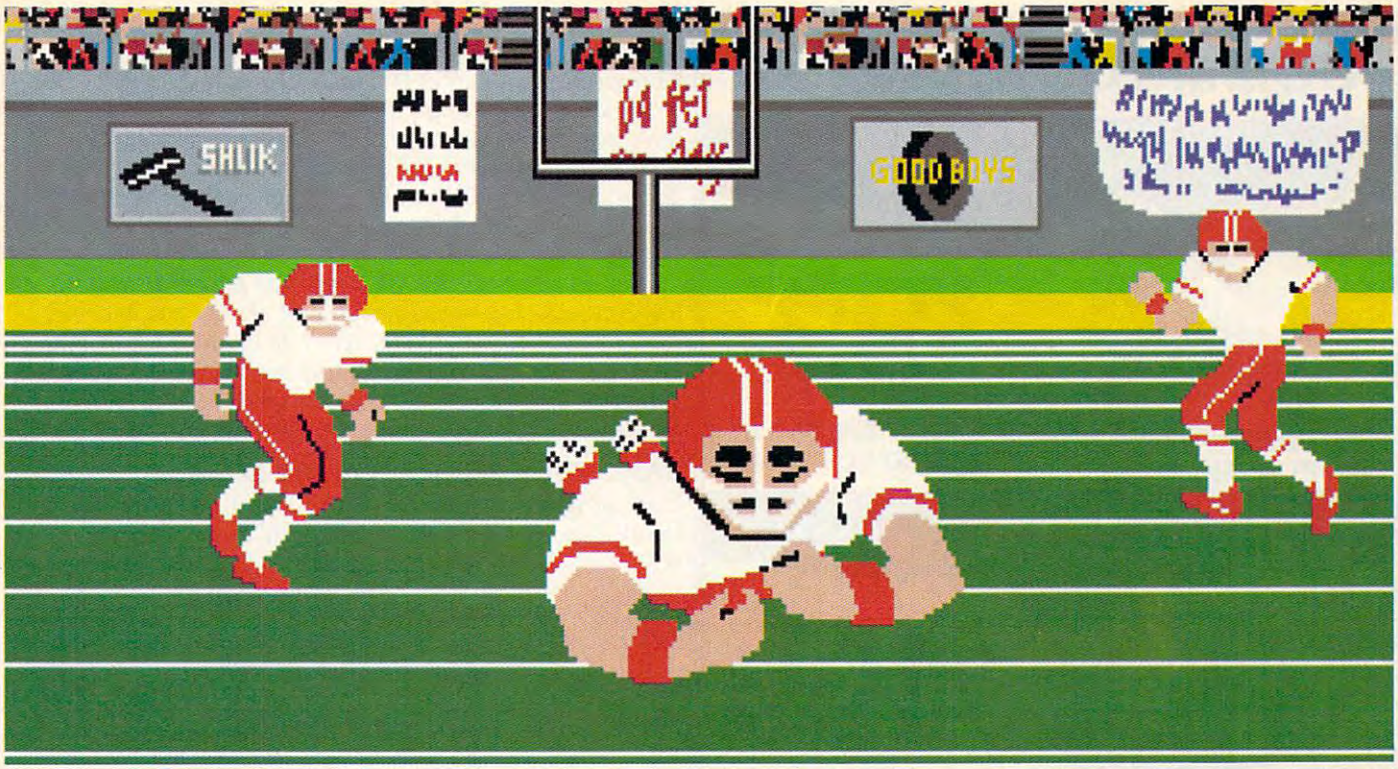
I am getting other numbers such as 12, 18, 36, and 63. What do those numbers mean?

Jamie Chulada

The mysterious numbers indicate that two or more sprites are colliding. The BUMP function reads the contents of the 128's sprite-collision registers and returns it to BASIC. BUMP(1) reports sprite-to-sprite collisions—the same as performing PEEK(53278) on the Commodore 64—and BUMP(2) reports sprite-to-foreground collisions—the same as PEEK(53279). Each bit of the collision register is assigned to one of the computer's eight sprites. When sprite 0 is involved in a collision, bit 0—the lowest bit of the collision register—is set to 1. When sprite 1 is involved in a collision, bit 1 is set to 1, and so forth. The second column of numbers that you list indicates the decimal values for each bit position in the collision register.

BUMP returns the sum of all the bits in the sprite-collision register. Thus, the number 12 indicates that sprites 2 and 3 are touching one another. Sprite 2's bit value is 4, and sprite 3's bit value is 8. When you add those two bit values together, you get 12. Similarly, the number 18 indicates that sprites 1 and 4 are colliding (2 + 16 = 18).

The BUMP value tells you which sprites are currently involved in some collision. But if more than two sprites are on the screen, it won't necessarily tell you which sprite is touching a given sprite or foreground object. For instance, the value



Amiga screen

SO REAL IT HURTS

GFL Championship Football™

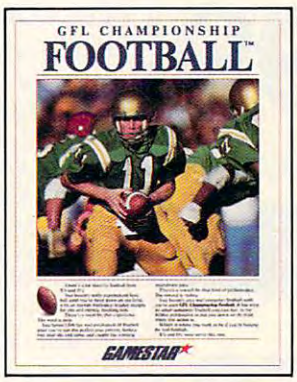
The way computer football *should* be.

Other football games put you in the grandstands, looking down on the action. Now see what it's like from the *player's* perspective—looking out of your helmet at an angry linebacker headed straight for you, and no blockers in sight.

With **GFL Championship Football,**™ you've got the first football simulation that actually takes you down on the field, taking the hits and making the plays. And it's more than just a pretty picture—you really get the feel of *playing* football.

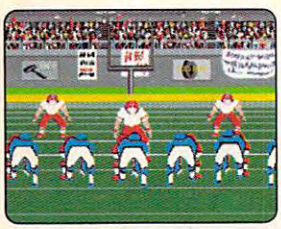
No other football simulation gives you so many features:

- *In-the-helmet perspective* puts you at ground level on the playing field.
- *Scrolling-screen animation* moves you up and down the playing field.
- *Realistic sound effects* let you hear everything from the quarterback calling the signals to the sound of your own footsteps.
- *Team selection screens* allow you to set the playing style of your team and that of your opponent.

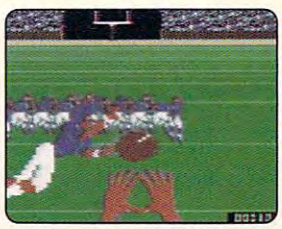


Whether you're taking on bone-crunching action against a friend, or going up against any of the 27 computer-controlled teams in the **GFL**, this is the one that puts you where the action is!

Available now for the Commodore 64/128, IBM PC and Tandy 1000, Apple II, Amiga, Atari ST and 100% compatible computers.

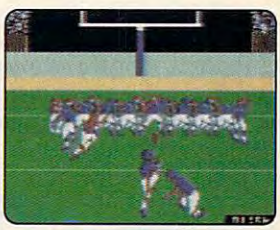


Amiga screen



Commodore 64-128 screen

Commodore 64-128 screen



Commodore 64-128 screen

GAMESTAR™

63 indicates that sprites 0–5 (1 + 2 + 4 + 8 + 16 + 32 = 63) are touching other sprites or foreground objects. But this result does not mean that each of those sprites is involved in the same collision as the others. For all you know, sprites 0 and 1 may have collided on one part of the screen, and sprites 2–5 may be involved in a three-way collision elsewhere.

In other words, BUMP(2) tells you that a given sprite has collided with some *foreground object, but does not indicate* which *foreground object it is touching. If that information is important, you must compare the horizontal and vertical screen positions of every sprite on the screen.*

BASIC Page Flipping On The ST

I am programming in ST BASIC, and I would like to know how to flip from one screen to another.

R. W. Sharples

Page flipping—switching from one display screen to another—is quite simple in a language like C, but it's not practical in ST BASIC. The first problem has to do with memory allocation. An ST screen requires 32,000 bytes, and it must begin at an address that's evenly divisible by 256. In order to use an alternate screen, you must reserve 32,000 bytes of memory at a location divisible by 256. This ordinarily would be done with GEMDOS routines, but ST BASIC provides no means to call a GEMDOS routine. If you attempt to use an unprotected memory area, you run the risk that output to the new screen will interfere with BASIC, or that BASIC's operations will corrupt the screen.

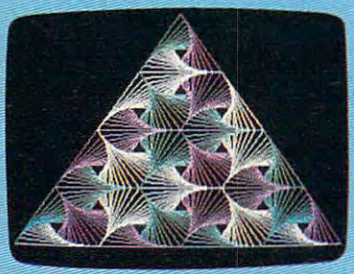
Assuming that you could surmount the memory problem, you also would have difficulty flipping from one screen to the next. Page flipping is done by calling an XBIOS routine, but ST BASIC also lacks any method for calling XBIOS routines. Furthermore, switching to a new screen requires that you pass to the system a 32-bit address representing the location of the new screen. Since the largest variable in ST BASIC is only 16 bits long, you have no practical way to tell the system where your alternate screen begins.

The recently introduced GFA BASIC *language permits access to GEMDOS and XBIOS routines, and it also has a built-in command that can flip screens without resorting to system calls. The SWAP command exchanges the values of two variables in* GFA BASIC. *If you previously have reserved a screen space named SCREEN2, this statement is all it takes to flip from the old screen to the new one:*

SWAP screen1, screen2 ©

GET UP TO 200 FUN-FILLED PROGRAMS EACH YEAR—

when you subscribe now to COMPUTE!



Subscribe to COMPUTE! today through this special introductory money-saving offer, and you'll be getting a lot more than just another computer magazine. That's because each issue of COMPUTE! comes complete with up to 20 all-new, action-packed programs.

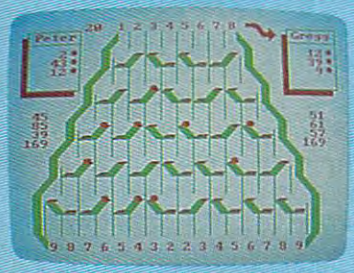
Subscribe now and you can depend on a steady supply of high quality, fun-filled programs like Hickory Dickory Dock, Switchbox, TurboDisk, Home Financial Calculator, Turbo Tape, SpeedScript, SpeedCalc, and hundreds of other educational, home finance, and game programs the entire family can use all year long.

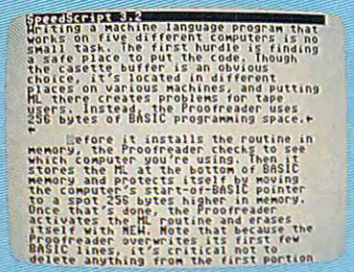
The superb programs you'll find in each issue are worth much, much more than the low subscription price.

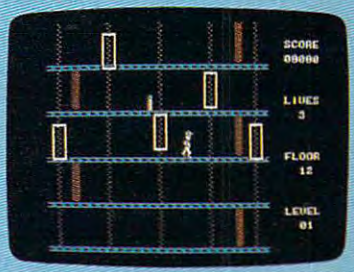
And there's more to COMPUTE! than just exciting new programs. Month after month, COMPUTE!'s superb articles deliver the latest inside word on everything from languages to interfaces...from programming to disk drives.

Whether you're a novice or an experienced user, COMPUTE! is the magazine for you. So subscribe today. Return the enclosed card or call 1-800-247-5470 (in Iowa, 1-800-532-1272).

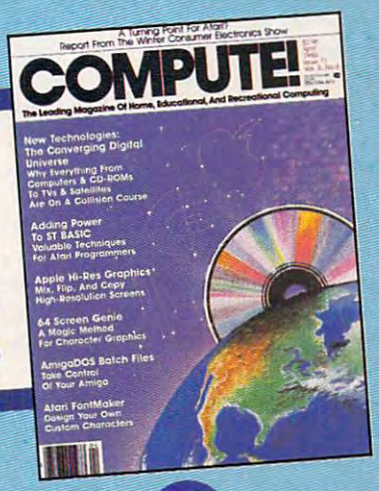
Do it now.







ACT NOW AND SAVE!



COMPUTE!™ Publications, Inc. abc

One of the ABC Publishing Companies

If attached order card is missing, write: COMPUTE! P.O. Box 10955, Des Moines, IA 50950

The New Music

Selby Bateman, Features Editor

Digital technology and computers are changing the ground rules of music. Sounds are being produced that have never before been heard. Many professional musicians are altering the business of commercial music by composing and performing in ways previously unthinkable. And nonmusicians can now create and play music with the help of smart computer programs that teach, guide, and accompany.

Pick up many of the latest records, tapes, and compact discs on the market and you're in for a surprise. In addition to the traditional credits given to those who play guitar or piano or drums, you'll increasingly find credit being given for *programming, digital mastering,* and other computer-related processes.

You may be in for a similar surprise at your next concert. One or two musicians can now play a bank of computer-controlled instruments that sound like an entire orchestra. Drum machines, sequencers, sound samplers, digital pianos, and synthesizers cover the stage—all hooked into one another and connected to one or more computers.

Musicians as varied as Frank Zappa, Philip Glass, Wendy Carlos, Jan Hammer, Vangelis, Steve Winwood, Pat Metheny, Peter Gabriel, and many others are experimenting with a variety of new-tech musical styles and machines as they explore the cutting edge of digital technology. More fundamental changes are occurring today in the ways we create, play, and listen to music than in any previous era. And those alterations are raising eyebrows, expectations, and problems.

For most computer users, however, the most direct effect of the changing musical landscape may be in the dozens of new and sophisticated music software packages that have been emerging over the past couple of years. Computers with more memory and power are providing a much richer environment for software developers, and this translates into some of the most accessible and flexible music programs ever developed.

The MIDI March

Each of these subject areas—sound generation, commercial production, and amateur access—is based on the revolution in music caused by the introduction of MIDI in 1982.

MIDI—the Musical Instrument Digital Interface—is a standard set of electronic specifications for interconnecting electronic musical instruments, and that includes computers. MIDI is both a hardware standard and a software standard, the basics of which were agreed upon by a number of the leading companies in the electronic music business, such as Yamaha, Sequential Circuits, Korg, E-Mu Systems, Roland, and others. The fact that these companies were able to agree on the standards back in 1982 has meant that all electronic music development could move forward much faster.

How important is MIDI? David Kusek, president of Passport Designs, a leading music software company, claims that MIDI is turning musical instruments into computer peripherals. "It's making it possible for a much larger group of people to make music," he adds. "MIDI is changing the nature of music learning and production."

The basics of MIDI are easy to understand. Let's say you have a personal computer, a synthesizer, a drum machine, and a sequencer. Before MIDI, it would have been virtually impossible to connect the four machines in any mutually productive combination. But through MIDI, you physically connect the four with cables and communicate via a common set of transmission signals that travel from machine to machine.

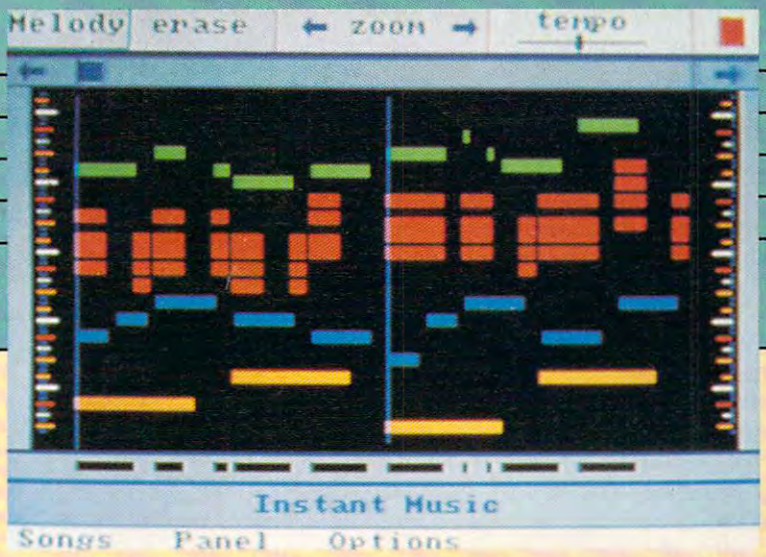
MIDI itself is an open-ended set of specifications, designating a minimum group of standards that all companies can follow. At the hardware, or machine, level, MIDI is really quite simple. MIDI ports can be MIDI IN, MIDI OUT, or MIDI THRU. MIDI IN ports receive the digital data, MIDI OUT ports send the data, and MIDI THRU ports pass along the data. The plugs, jacks, and cables used by MIDI must be the same. The cable is the common shielded, twisted-pair type, and the ports are the standard five-pin DIN variety.

There are 16 separate MIDI channels that can be set to send, carry, or receive data from different instruments. In the newest instruments, individual voices can be assigned to different channels. They operate in much the same way that television channels do, but the sending and receiving options are much more flexible and interactive with MIDI channels. There are also a variety of modes for sending and receiving information. As you can see, at its most basic level, MIDI is very simple; at higher levels, with many machines interconnected and different channels carrying different voices, the results can become both complex and powerful.

The New Professional Environment

For most computer users and amateur musicians, however, there's no real need to become a technical wizard to exploit the promise of MIDI. Some new computers, like the Atari ST, come with MIDI ports already installed. And MIDI interfaces for personal computers are getting much cheaper and more versatile.

For professional musicians, there's every reason to explore the



The computer becomes your musical accompanist and teacher with Instant Music, *from Electronic Arts.*

many uses of MIDI. The results among musicians who have already become proficient with computer-aided, MIDI-controlled composition and performance have been remarkable. Most dramatic, perhaps, are the works of composers such as Jan Hammer, who every week single-handedly scores an hour-long episode of the television program "Miami Vice" from the computer-controlled recording studio in his home. In a similar fashion, the composer Vangelis created, by himself, the entire award-winning score for the movie *Chariots of Fire*, composing and producing all of the music.

Frank Zappa—who has, in the past, delighted in writing musical compositions too difficult for musicians to play—now has digital music machines that can do the job quite easily. "I use synthesizers for three things," says Zappa. "For generating sounds that never existed before, for performing music which human beings would have difficulty playing, and to get rid of some of the drudgery of composition."

While professional musicians may be more experienced in composing and performing music, their goals are not unlike those of non-musicians who want to make music. And thanks to a new breed of music software, amateurs today can do more and sound better than ever before.

Improving Hardware

Making music on a computer has come a long way in a very short time. Before computer manufacturers put music chips in their computers, some adventurous computer users made sound by actually programming their computers to tell their printers to tap out meager rhythmic patterns. The first sound-producing computers used simple tone generators with oscillators that could affect pitch and volume, and not much more.

For several years, the Atari eight-bit computers' four-voice sound chip was the best that could be had on a personal computer. But then came the Commodore 64's amazing SID (Sound Interface Device) chip which—five years later—is still a remarkable sound processor.

But the greatest leap has been in the advances in sound-generation capabilities that have come with the latest generation of computers. Add to that the vastly expanded power that these computers have because of their 512K and even one-megabyte memories, and the musical landscape looks even broader.

The Amiga's four-voice stereo sound output, with independently programmable volume level and sound-sampling rates, is only now beginning to be effectively tapped. And the Apple IIGS computer goes even further in sound generation with the amazing Ensoniq Q chip that has 15 separate, two-oscillator voices and a built-in analog-to-digital converter. It will take a while before software developers exhaust the musical power of the Amiga and Apple IIGS computers.

At the same time, both the Macintosh and the Atari ST computers are attracting professional and amateur musicians alike to their powerful and yet easy-to-use environments. During the past couple of years, software developers have produced quite an array of music-composition programs for the Macintosh, and the same situation seems likely for the ST. In fact, Atari engineers realized that the potential for musical applications of the ST was so great that they designed MIDI IN and MIDI OUT ports on the back of the STs when they were first built. So, instead of needing a MIDI interface to connect between the computer and MIDI instruments, the ST is already set for MIDI use.

More Memory, More Music

There has developed a very large library of music software for eight-bit machines like the Commodore 64 and the Apple II–series computers. And many professional musicians first began tinkering with digital music, MIDI, and computers on one of these eight-bit machines.

But despite the flexibility of these computers, the pros soon found themselves reaching the limit of memory on the 64K machines. It's possible to get about 6000 notes into memory at one time on a Commodore 64. And if you want to process those notes in any advanced ways—say, by pitch bending or using a modulation wheel on a MIDI-equipped synthesizer—the memory is used even faster.

The new-generation computers, with 512K or as much as one megabyte of memory, can handle virtually all of the notes and processing that even the most demanding composer can throw at them. Software companies have not been slow to realize this potential. Activision, Aegis Development, Cherry Lane Technologies, Dr. T's Music Software, Electronic Arts (EA), Hybrid Arts, MidiSoft, Passport Designs, Sonus, and Southworth

Music Systems are but a few of the companies that have produced a number of music software programs for both professionals and amateurs. (See accompanying music buyer's guide.)

There are almost as many kinds of music software available today as there are packages. But most of them fall into one of three broad categories: educational programs aimed at systematic teaching, training, and/or practice of musical knowledge and skills; entertainment software aimed at unleashing the creative and playful aspects of music creation and performance while also allowing some level of serious productivity; and MIDI-related programs that serve as controllers for you to use with your computer and one or more MIDI-equipped musical instruments.

Many of the educational programs have proven to be a boon to music instruction in school settings as well as in the home. But it's the latter two categories—the creative programs and the MIDI programs—that seem to be capturing the fancy of most amateur and professional musicians. In addition, an increasing number of the newest music-creativity programs are being developed with MIDI compatibility already built in.

The range of options and features that are a part of most MIDI programs—sequencers, editors, controllers—is remarkable. Passport Designs' new *Midisoft Studio* for the ST, for example, is a complete multitrack recording studio and sequencer that features real-time recording, playback, overdub, rewind, and fast forward. It has 32 polyphonic tracks which are independently controlled, and a capacity for more than 80,000 notes per song. In addition, there is full track editing for combining, moving, copying, and erasing any combination of the 32 tracks. In other words, you can change virtually any musical parameter you can think of in just about any manner.

For computer users who aren't interested in using their machines with electronic synthesizers, drum machines, digital pianos, and the

Activision's The Music Studio *is an entertaining creativity program which also has a full set of music-composition tools.*

like, there are plenty of software programs that use just the computer to compose and perform music. Among the best-known and the most complete of these programs for both 8-bit and new 16-bit computers are Activision's *The Music Studio* and Electronic Arts' *Music Construction Set* (and the new *Deluxe Music Construction Set*).

The Music Studio, for example, is something of a musical tool kit that has an impressive array of features, but is also accessible to beginners. The program offers full composing capabilities, as do many programs, but there are also tools for creating your own instruments and sound effects, and a "paintbox" feature for free-form musical experimentation. Activision also offers MIDI capability on the ST, Amiga, Tandy 1000, and Commodore 64 versions.

In a similar fashion, *Music Construction Set* and the new deluxe version offer free-form composition tools and user-definable sounds. The emphasis in both programs is to give the beginner plenty to play around with and to enjoy, without having to know too much at the start. As the level of knowledge and skill goes up, the programs have built-in tools that are quite sophisticated.

Your Computer Accompanist

A most interesting offshoot from these composition and entertainment programs are software packages that actually become accompanists to your creative and performance efforts. This is the logical next step, and one that promises to bring even more non-musicians into the computer-music fold.

One of the newest and best examples of this breed of helpful music software is Electronic Arts' *Instant Music*, a program that won't let you make a mistake—unless you want to. The software does this by keeping you in the right key and rhythm no matter what you're playing. You can even "Mousejam" along with the program—using the mouse to control one instrument as several other instruments play a composition. No matter where you move the mouse on the musical staff, you're in key and in rhythm and always following the melody. The computer becomes your musical partner. For a nonmusician, the experience is both fascinating and educational.

Instant Music, and a few programs like it, provide that one extra step that can help a beginner really get excited about creating music. "*Instant Music* is a result of what we learned from *Music Construction Set*," says EA producer Stewart Bonn. "Although we had freed a person from having to play a keyboard in order to play music, we hadn't necessarily taught them where to place the notes. And, unfortunately, music composition is composed of a lot of rules that not a lot of people understand.

"*Instant Music* lets the computer take care of all those rules," he says. "It's as though you had the computer holding your hand and guiding you."

It's clear that the digital-music invasion is just underway. And computers will remain in the forefront of this amazing musical transformation. The digital music machines and computers that you can buy today for less than a thousand dollars can produce far more sophisticated results than musicians could have achieved 20 years ago in a firstrate recording studio. And much of that power comes from MIDI.

Says one music-software developer, "The real power is with the consumer. MIDI will allow the marginal musician perfect performances, if he's willing to use it."©

TEST DIVE ONE FOR YOURSELF.

In their day, they ruled over three quarters of the earth's surface.

During WWII, they viciously brought Britain to her knees. And Japan to the ground.

These were the silent killers: Tench. Gato. U-Boat.

And now, they return. In this, the most realistic, all-encompassing simulation ever created for the personal computer.

You will command one of six types of American subs or German Kriegsmarine U-Boats, during any year from 1939 to 1945. You'll perform one of over 60 missions. Or you'll engage in the most difficult task of all: To make it through *the entire war.*

Each vessel is completely unique and painstakingly authentic, so you'll have a lot to learn: Navigation. Weather. Radar.

TAKE OUR PREVIEW DISK FOR A SPIN. Drop this coupon in the mail with your check or money order, and we'll gladly send you to the South Pacific to have it out with an enemy fleet.

Mail to Sub Battle Preview, P.O. Box 8020, Redwood City, CA 94063.

	Quantity	Total
Apple II & compat. (128K)	_____	$1.50 ea. _____
Commodore 64/128	_____	$1.50 ea. _____
IBM PC & compat.	_____	$1.50 ea. _____
Atari ST	_____	$2.75 ea. _____
Macintosh	_____	$2.75 ea. _____
Apple II GS	_____	$2.75 ea. _____
Total Disks Ordered	_____	Total Enclosed _____

Name _____ Phone ( ) _____

Address _____ Age _____

City/State/Zip _____

Canadian orders please add 50¢ for additional postage.

Please allow 4 to 6 weeks for delivery. Offer expires 8/30/87 and is valid only in the continental U.S. and Canada. Void where prohibited.

30

And the contents of a vital target book, among other things.

Your arsenal will include deck and anti-aircraft guns. Torpedoes. And mines.

But even all that may not be enough.

Because besides the risk of bumping a depth charge or facing a killer Destroyer, you'll still have to contend with the gunfire of enemy aircraft.

No simulation has ever had the degree of authenticity, gut-wrenching action or historical accuracy of this one.

The first release of our new Masters Collection. And a challenge of unbelievable depth.

EPYX®

Apple II & compatibles, Apple IIGS, Atari ST, C64/128, IBM & compatibles, Macintosh.

The No. 1 battery. The ship's heart.

Sea guard radar stub.

Your ammo.

The 360° periscopes.

The sealed control room. Your HQ.

Independent generator & diesel engines.

Salt water tank, for trimming and compensating.

5" 25 cal. gun.

Officer's quarters.

Water purification.

Glossary Of Electronic Music Terms

Amplitude—loudness.

Analog sound—recordings on ordinary tape recorders or vinyl records. The sound waves on these media replicate the waves which will hit the air when the tape player or record player is turned on. You can see them if you look closely at an LP: little fluctuations in the grooves which are an *analogy* of the sound therein contained.

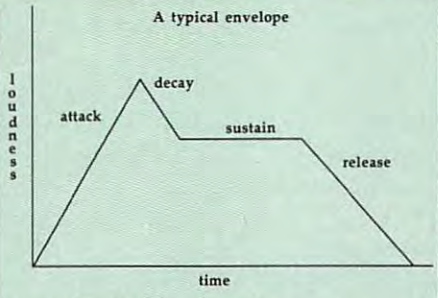
Bandwidth—the amount of fidelity. The distance between the lowest and highest frequencies possible in a given instrument or device.

Digital sound—recordings on compact disc or digital tape. The sound waves bumping against a microphone are translated into *numbers* (digits) which are then stored. Sound information stored in this fashion is far less susceptible to the dust, warpage, and other kinds of decay which have plagued analog storage media since their invention in the nineteenth century. More importantly, the numbers can be easily *processed* at virtually no cost. If you want to add echo, just copy the pitch numbers, adjust the timing numbers, and reduce the loudness numbers associated with the copy. All this is a software event in the digital domain: Nothing physical has to happen, just some math. Contrast that to the expensive electronics required to send analog music through a device that has to somehow physically control the necessary repetitions and relationships.

Dynamics—variations in amplitude.

Envelope—how the sound builds and dies away. Broken into four fundamental phases—attack, decay, sustain, and release—the envelope of a sound is the variation in its amplitude over time.

The ear is very sensitive to variations in the loudness of a



sound, and the envelope is one of the most important ways that we distinguish different musical instruments. Some instruments have similar waveforms (pitch relationships) but are easily told apart because one abruptly goes silent while the other slowly fades.

Fidelity—how well a recorded or synthesized sound matches the original. For instance, a two-inch speaker will always be low fidelity no matter how good a signal you feed into it. It's just too small; few musical instruments have two-inch openings through which their sounds normally pass. Forcing the big boom of a bass drum through a two-inch opening is a doomed endeavor: The sound waves are just too large to fit through, and such a speaker is politely described as "lacking in bass." Attach larger speakers to the system, though, and you'll get higher fidelity.

Filtering—selectively removing elements of a sound. When you turn down the treble control on your stereo set, you are filtering out some of the high-frequency content of the music. It sounds less bright because you are invoking a variable filter which eliminates a portion of the sound.

Low-pass filter—a device which allows the low-frequency content of a sound to pass through, but blocks the high-frequency content. In digital recording, there are effects beyond the range of human hearing which nevertheless can distort the sample and which require low-pass filtering. Such filtering is also used to eliminate hiss or other high-frequency noise.

Noise—disorganized sound. Noise can come from the 60-cycle-per-second hum of ordinary electrical current if electronic equipment isn't properly grounded, from the hiss caused by imperfections in recording tape, or from other sources such as inadequately shielded computer circuitry. Whatever its source, noise is a constant problem in the creation of music and its high-fidelity reproduction. Tape hiss might well be the exact same sound as a brushed cymbal, but the cymbal is brushed with the music, on the beat, while the tape noise is random.

Orchestration—the choice of instruments. Deciding, for example, that you want your melody played by a clarinet and not by an oboe is orchestration.

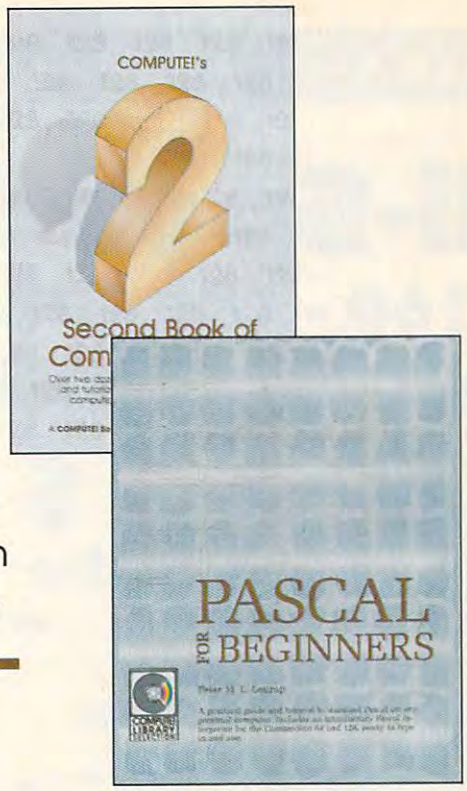
Oscillator—an electronic device which vibrates, causing electrical signals to take on waveforms. Useful in generating sound.

Pitch—how high or low a sound is.

Polyphonic—more than one sound at a time. A soloist singing a melody without accompaniment is *monophonic*. But when you add a guitar, a drum, and other musicians, you get polyphony. An important aspect of a musical instrument is the number of sounds it can make simultaneously. A drum is normally monophonic, but a set of drums can be played polyphonically.

Reverb—complicated clusters of echoes which add fullness and naturalness to a sound and which are caused by reflections of sound waves off the walls of a room. Differences in reverberation are what you hear when you can distinguish the sounds made by the same piano

New books from COMPUTE!



COMPUTE! Books is bringing you a brand new line up of books for your Commodore 64 and 128. These recent releases offer you everything from programming hints to exciting games, from educational to home and business applications.

Pascal for Beginners

$14.95 0-87455-068-8
Book/disk combination for the Commodore 64
$29.95 ISBN 0-87455-069-6
This introductory text to standard Pascal on any computer is an ideal tutorial for anyone who wants to learn this powerful computer language. It includes everything you need, including an introductory Pascal interpreter* for the Commodore 64 and 128 in 64 mode, ready to type in and use. Written in plain English and offering numerous program examples, it gently and clearly explains standard Pascal and structured programming. Latter sections include discussions of advanced topics such as files and dynamic data storage. *There is also an optional disk available for $12.95 for the Commodore 64 which includes most of the programs in the book. 688BDSK.*

*The Commodore 64 Pascal interpreter is not full-featured, but still a powerful implementation of Pascal which suits the needs of most beginners.

COMPUTE!'s Music System for the Commodore 64 and 128

Book/disk combination only
$24.95 ISBN 0-87455-074-2
Sidplayer, the feature-packed, popular music player and editor program, is now more versatile and more impressive than before. *Enhanced Sidplayer* for the Commodore 128 and 64 includes two new versions—one for the Commodore 128 running in 128 mode and another for the Commodore 64. Take advantage of every feature the SID chip (the sound chip in the 128 and 64) has to offer. Just like the original, *Enhanced Sidplayer* is easy to learn and use, with many powerful new features. The accompanying disk contains the editor, player programs (including a Singalong program), utilities, and sample music that you can enjoy immediately or change. The new *Sidplayer* plays any songs created by the original *Sidplayer* for the Commodore 64.

User's Guide to *GEOS*: geoPaint and geoWrite

$18.95 ISBN 0-87455-080-7
Learn the ins and outs of *GEOS*, the new icon-based operating system for the new Commodore 64C and the 64, with this step-by-step guide. Everything from creating simple letters with *geoWrite* and pictures with *geoPaint* to merging text and graphics and using desk accessories is clearly and concisely explained.

COMPUTE!'s Second Book of the Commodore 128

$16.95 ISBN 0-87455-077-7
The editors at COMPUTE! Publications have collected some of the best games, programs, and tutorials for the Commodore 128 (in 128 mode) from *COMPUTE!* magazine and *COMPUTE!'s Gazette.* Like *COMPUTE!'s First Book of the Commodore 128,* this book offers a variety of programs and articles for every 128 user. Each program has been fully tested and is ready to type in and use on the Commodore 128 running in 128 mode. *There is also a disk available for $12.95 which includes the programs in the book. 777BDSK.*

Mapping the Commodore 64, Revised

$16.95 ISBN 0-87455-082-3
An update of the bestselling memory map and programming guide. It's a necessity for intermediate and advanced programmers. This definitive sourcebook has been expanded and now covers the new icon-based *GEOS* (Graphics Environment Operating System) with clear descriptions of how to make it work for you. For BASIC and machine language programmers of both the Commodore 64 and 64C.

Look for COMPUTE! Books at your local computer or book store.
Or, to order directly from COMPUTE!, call toll free 1-800-346-6767 (in NY 212-887-8525) or write COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Please include shipping and handling: $2.00 per book in U.S. and surface mail; $5.00 airmail.
NC residents add 5 percent sales tax and NY residents add 8.25 percent sales tax.
Please allow 4–6 weeks for delivery.

COMPUTE! Publications, Inc. abc
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies

COMPUTE! Books are available outside the United States from subsidiaries of McGraw-Hill International Book Company.

SALE • *Famous National Brand* **• SAVE**

NLQ 180
Hi-Speed Printer Sale

•160 - 180 CPS • Near Letter Quality •

Lifetime Warranty*

Price Break Thru

Sale $199.00 List $499.95 **Fantastic Price**

60% OFF LIST PRICE

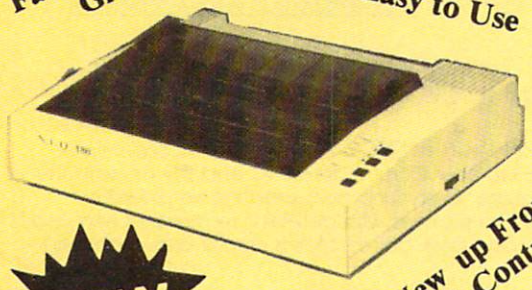
Below Wholesale Cost Prices!!!

NLQ-180 Premium Quality Printer

10" Carriage

Fantastic Graphics

Easy to Use



All New up Front Panel Controls

NEW

Lifetime Warranty*

Near Letter Quality Selectable From Front Panel Controls • High Speed Dot Matrix• Letter Quality Modes • 8K Buffer frees up computer 4-times faster • Pica, Elite, Italics, Condensed • Super Graphics • Business or Personal • Tractor/Friction • 15 Day Free Trial • Lifetime Warranty on Print Head* • 6 Month Immediate Replacement Warranty •

NLQ-180 Print Samples

This is an example of ITALICS
Enhanced Boldface
Condensed Text Double-strike
example of Near Letter Quality

— IBM — COMMODORE — EPSON — NLQ180 SPECIFICATIONS ——— APPLE — ATARI — ETC. ———

Print Buffer
8K bytes utility buffer
Printing Direction
Text Mode — Bi-directional
Graphic Mode — Uni-directional
Interface
Centronics type parallel (8-bit)
Paper
Plain paper, Roll paper, Single sheet
Fanfold, Multipart paper: max. 3 sheets (original plus 2 copies)
Character Fonts
Pica, Elite, Italics, Condensed

Printing Method
Impact dot matrix
Printing Speed
160-180 CPS at standard character printing
Printing Characters
Standard 9 x 9 dot matrix
NLQ 12 x 18 dot matrix (33cps)
Character size: 2.12 x 2.8 mm (standard)
Character sets: Full ASCII character set (96)
32 International characters

Ink Ribbon Cartridge
Ribbon Life: 3 million characters/cartridge
Physical Dimensions
Size: 15" x 12" x 5"
Weight: 12.7 lbs.
Maximum Number of Characters

Standard:	10 cpi	80 cpl
Standard enlarged:	5 cpi	40 cpl
Elite:	12 cpi	96 cpl
Elite enlarged:	6 cpi	48 cpl
Condensed:	17 cpi	132 cpl
Condensed enlarged:	8.5 cpi	66 cpl
Condensed elite:	20 cpi	160 cpl

INTERFACES

Atari $39.95 Apple $49.95 Commodore $29.95 IBM $49.95 Laser $19.95

Add $10.00 for shipping, handling, and insurance. Illinois residents please add 6¼% sales tax. Add 20.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, APO-FPO orders. All 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. C.O.D. on phone orders only.

COMPUTER DIRECT

22292 N. Pepper Rd., Barrington, Illinois 60010

Call (312) 382-5050 or 382-5244 to Order *We Love Our Customers*

Twice the speed* at just a fraction of the cost!

* Computes over two times faster than the IBM® XT

10 MHz Super Turbo IBM® XT Compatible Computer System

Run thousands of IBM® software programs available.

Look at all you get for only $599.00

The complete system	List Price	Sale Price
10 MHz Super Turbo XT Computer	$1295.00	$499.00
* 512K Memory	$99.95	No extra cost
* Single floppy disk drive	$129.95	No extra cost
* Parallel printer port	$59.95	No extra cost
* Serial printer port	$59.95	No extra cost
* Mouse/joystick port	$59.95	No extra cost
* RGB color graphics port	$99.95	No extra cost
* Hercules compatible monochrome port	$79.95	No extra cost
MS DOS 3.2 & GW Basic	$199.00	$99.00
12" Hi-Res 35 MHz Green Screen Monitor (TTL & EGA compatible)	$249.00	$99.00
Monitor interface cable	$24.95	$19.95
Big Blue Printer	$199.00	$39.95
RS 232 IBM to Big Blue cable	$19.95	$9.95
2 rolls of paper	$19.95	$5.95
Word First • Word Processor	$99.00	$39.95
Data First • Data Base	$99.00	$39.95
Calc First • Spreadsheet	$99.00	$39.95
Total price when bought separately	$2893.85	$892.65

Home & Business

This IBM® XT compatible is perfect for your home and/or business uses. It makes life easier in more ways than you can imagine. Use the system for personal letters, form letters, address storage, listing valuables, figuring finances, school reports, business reports, calculations, business projections...the list can go on and on. With the addition of some of the thousands of software programs available for IBM® you can increase the capabilities of your system even further. A terrific home improvement, business enhancer, entertainment center & educational aid!

Save over $275.00 off sale prices!

Complete System only $599

* Built-in the Super Turbo XT

IBM® is the trademark of International Business Machines Inc.

* 90 Day Immediate Replacement Policy from Computer Direct

Shipping, Handling & Insurance Charges and Information

Add $35.00 for shipping, handling and insurance. Illinois residents please add 6½% sales tax. Add $70.00 for CANADA, PUERTO RICO, HAWAII, ALASKA and APO-FPO. All orders must be in U.S. dollars. 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.

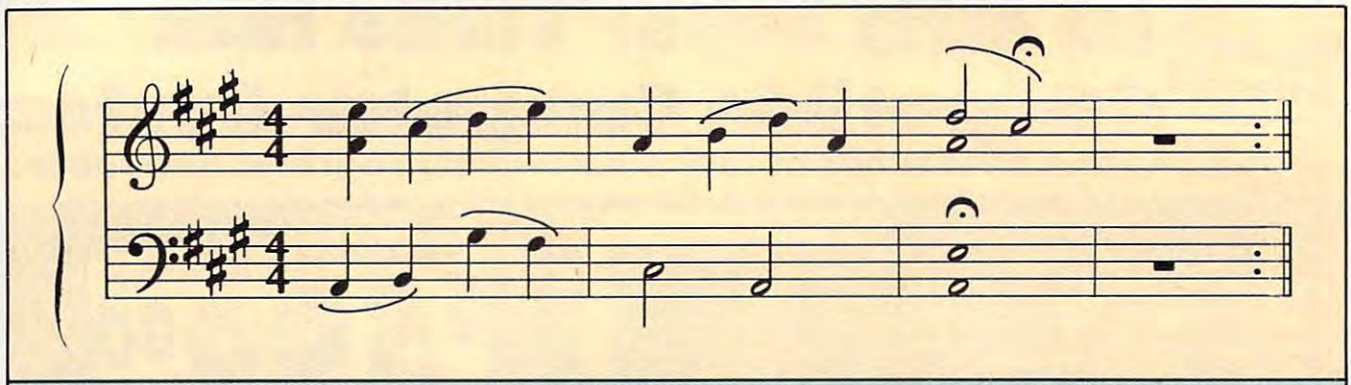
VISA - MASTERCARD - C.O.D. Please call for C.O.D. charges.

COMPUTER DIRECT

22292 N. Pepper Rd., Barrington, Illinois 60010

Call (312) 382-5050 or 382-5244 to Order

We Love Our Customers



played in a small room or played in Carnegie Hall. Reverb is often added artificially in recording studios to make music sound more real and more pleasing to the ear.

Ring Modulation—a special way of superimposing filtering on a sound which results in the exotic, constantly shifting timbre characteristic of bells and gongs.

Sampling—making a brief digital recording of a sound or musical instrument. Although we're all familiar with *analog recording* using tape recorders, the quality of digital recording can be much greater; and the resulting sound, captured in RAM memory instead of tape, is far easier to work with. For example, if you sampled the sound of a pencil hitting a cup, you could then play the sound back at different pitches, as if you had 88 cups sitting inside a piano, 1 for each key. Or you could modify the sound in a variety of ways (echo, play it in reverse, and so forth), which is quite easy to do when the sound resides in computer RAM memory, but difficult, if not impossible, when it's on tape. Sampling, however, does use up RAM memory very quickly. A few seconds of sampled sound can require thousands of bytes of storage space.

Sampling Rate—how often, per second, the sound waves hitting a microphone are measured and transformed into numbers. All things being equal, the higher the sampling rate, the more the resulting sample will resemble the original.

Sonic—pertaining to sound.

Sync—using one oscillator to control another to produce such effects as tremolo (where the pitch rapidly rises and falls, almost like yodeling) or vibrato (where the amplitude rises and falls).

Synthesis—creating artificial sounds from scratch. Using the elements of sound (waveforms and envelopes), it is possible to build very close approximations of acoustic instruments or to invent entirely new sounds.

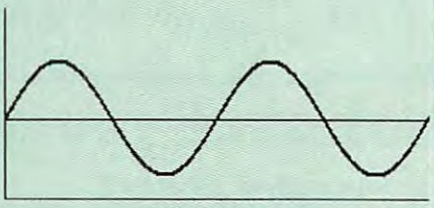
Sound is vibration. It's a disturbance of the air that forms wave-like patterns which strike the ear. And there are two fundamental elements to sound: pitch and amplitude. Pitch is how high or low the sound is on the musical scale and is a direct result of how many vibrations per second are occurring. A high pitch is caused by frequent vibrations; a low pitch by fewer vibrations. The amplitude is how loud the sound is.

Synthesists can use electronics and computers to create waveforms and control amplitude in complex ways. They can superimpose, invert, filter, and otherwise manipulate them into sounds that are *designed* rather than *natural*. Modern music is becoming increasingly reliant on synthesis in the same way and for the same reasons that modern manufacturing increasingly relies on synthetic materials: The product is often less expensive, more reliable, and, sometimes, cannot be found in nature.

Transpose—applied to digital sampling, this means to move a sound up or down in pitch. A drum transposed up three octaves could sound like a bird chirp—a shorter and higher-pitched sound. Transposing a sampled sound so far from its normal range is called the *Mickey Mouse effect* because the sound begins to take on an odd, hollow quality. For this reason, several different samples of instruments with wide pitch ranges (such as the guitar) need to be made. The piano, one of the most difficult instruments to sample, requires many samples across its range. The waveforms of the low notes and high notes on a piano are so distinct that they seem to derive from different instruments altogether.

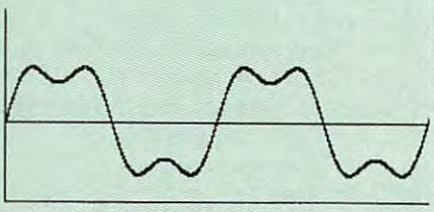
Voicing—adding character to a sound. Changing the voicing of an organ can make it sound like reed or wind instruments, for example.

Waveforms—The sometimes intricate shapes of the sound waves characteristic of different sounds. The sine wave (shown below), the simplest waveform, sounds like a flute.



A sine wave

If you start to deform the waves, like this:



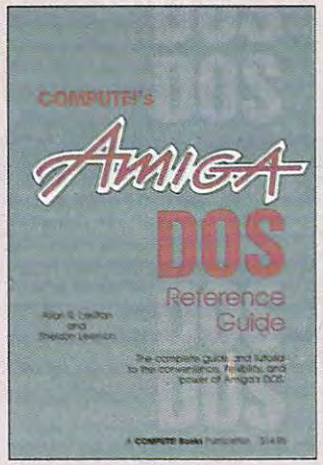
A modified sine wave

you'll start hearing a more raspy sound. Enough deformation, and you can end up with what sounds like a trumpet. Manipulating waveforms, in combination with control over a sound's envelope, can produce the sound of any instrument. The unique quality of an instrument's sound, its particular waveform, is called its *timbre*. ©

Amiga

SUPPORT FROM COMPUTE! BOOKS

Everything for the Amiga. From BASIC beginner's guides to advanced programming handbooks, COMPUTE! offers you information-packed tutorials, reference guides, programming examples, ready-to-enter applications, and games to help you develop your computing skills on Commodore's Amiga.

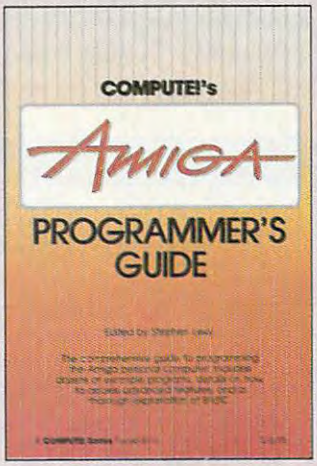


COMPUTE!'s AmigaDOS Reference Guide

Arlan R. Levitan and Sheldon Leemon
A comprehensive tutorial and reference guide to the powerful AmigaDOS—the operating system underlying the Workbench and Intuition—this book offers information useful to every Amiga owner. It defines and illustrates all DOS commands, and shows you how to create file directories, access peripherals, run batch file programs, and avoid "disk shuffle." The screen- and line-oriented text editors are explained in detail. Numerous examples and techniques explain how to use AmigaDOS to make operating your Amiga both convenient and efficient.
$16.95 ISBN 0-87455-047-5

Elementary Amiga BASIC

C. Regena
Here's your introduction to the new and powerful BASIC on the Amiga personal computer. The Amiga's impressive graphics, animation, and sound can be unlocked with the right commands, and BASIC is the place to start. Complete descriptions of Amiga BASIC's commands, syntax, and organization take you from the beginner level to a full-fledged programmer. Plus, the book offers you ready-to-type-in programs and subroutines while showing you how to write your own programs. *There is a disk available which includes the programs in the book,* $12.95. *This title is also available as a book/disk combination for $29.95* (057-2).
$14.95 ISBN 0-87455-041-6

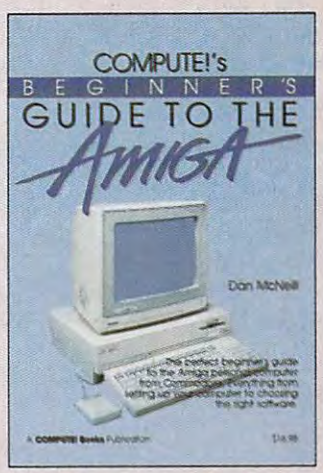


COMPUTE!'s Amiga Programmer's Guide

Edited
Your tutorial and reference manual to AmigaDOS, BASIC, Intuition, and other important software tools which accompany the new Amiga, *COMPUTE!'s Amiga Programmer's Guide* is a clear and thorough guide to the inner workings of this fascinating new-generation computer. The great speed of its 68000 microprocessor, coupled with the versatility of the Amiga-specific graphics and sound, makes the Amiga one of the most powerful computers available today.
This book is the key to accessing the Amiga's speed and power.
$17.95 ISBN 0-87455-028-9

Advanced Amiga BASIC

Tom R. Halfhill and Charles Brannon
This guide to applications programming on Commodore's new Amiga contains everything an intermediate programmer requires to begin creating sophisticated software on this powerful machine, including several ready-to-type-in programs. Clear, yet comprehensive documentation and examples cover advanced BASIC commands, designing graphic applications, generating sound and music, using the Amiga's built-in speech synthesizer, creating a user interface, and programming the computer's peripherals. *There is a disk available which includes the programs in the book,* $15.95. (June release)
$17.95 ISBN 0-87455-045-9



COMPUTE!'s Beginners Guide to the Amiga

Dan McNeill
Written in a lively and entertaining style, this book teaches you everything a beginner needs to know to get started quickly with the Amiga from Commodore. You will learn about setting up the system, all the most popular types of software, and details about the hardware.
$16.95 ISBN 0-87455-025-4

Inside Amiga Graphics

Sheldon Leemon
The Amiga, Commodore's powerful new computer, is an extraordinarily impressive graphics machine. Easy to use, the Amiga can produce color graphics and excellent animation. You'll find thorough descriptions of the computer's abilities and the hardware required to create a complete graphics system. Software, too, is central to the Amiga's power, and complete tutorials show you how to get the most from the machine. (June release)
$17.95 ISBN 0-87455-040-8

COMPUTE!'s Kids and the Amiga

Edward H. Carlson
The latest in this bestselling series written by Edward Carlson, *COMPUTE!'s Kids and the Amiga,* will acquaint you with BASIC. Over 30 sections—all with instructor notes, lessons, assignments, and lively illustrations—entertain and amuse you as you learn to program your new computer. Clear writing and concise examples make it easy for anyone—children and adults alike—to painlessly learn BASIC. (May release)
$14.95 ISBN 0-87455-048-3

Look for these books at your local book or computer store.
Or order directly from COMPUTE!.
Call toll-free 1-800-346-6767 (in NY 212-887-8525).

Please allow 4–6 weeks for delivery after your order is received.

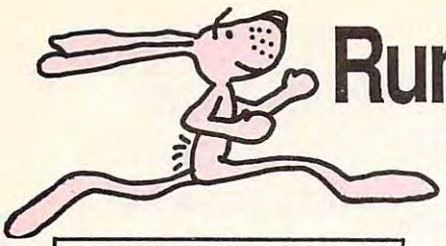
COMPUTE! Publications, Inc.
Part of ABC Consumer Magazines, Inc.
One of the ABC Publishing Companies
825 7th Avenue, 6th Floor, New York, NY 10019
Publishers of COMPUTE!, COMPUTE!'s Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications

COMPUTE! books are available outside the United States from subsidiaries of McGraw-Hill International Book Company.

A Buyer's Guide To Music Software

The programs listed here are only some of the hundreds of music software packages available for personal computers. This buyer's guide is not meant to be exhaustive, but does give you some idea of what's available and which companies are producing music software. A number of the companies mentioned here have a variety of other music programs available. The following guide does not attempt to include the professional programs priced significantly above the general consumer level. Note that prices and machine availability change frequently.

Product	Price	Publisher/ Vendor	System	Description
Adams' Music Disk Version 6.0	$39.95	Adams' Soft	Apple II, IIe, IIc	Elementary music-learning program with colorful graphics. Most useful for elementary-school teachers.
Advanced Music System	$79.95	Firebird	Commodore 64/128	A music program allowing creation of full compositions with MIDI capability. Suitable for the professional musician as well as beginners.
Aegis Sonix	$79.95	Aegis Development	Amiga	Create any type of music by combining multiple instruments and sounds with this professional music-composition program. An expanded version of a program originally called *Musicraft*.
Bank Street Musicwriter	$49.95	Mindscape	Apple II+, IIe, IIc; Atari eight-bit; Commodore 64; IBM PCjr	Composing comes to life as you arrange music on the screen. It's as easy to learn as arranging words in word processing.
Basic Chords	$39.95–$99.95	Electronic Courseware	Apple II+, IIe, IIc; Commodore 64/128; IBM PC, PCjr; Tandy 1000	Computer plays a basic chord or its inversion, which the user must then identify.
Basic Guitar 1	$50	Digital Concept Systems	Apple II, II+, IIe	Two-disk set of sound and graphics to teach chords to beginning guitarists.
Basic Piano Theory Software	$29.95	Alfred Publishing	Apple II+, IIe, IIc; Commodore 64/128	Creative graphics and animation in game formats reinforce concepts taught in Alfred's Basic Piano Theory.
Beatles Classics	$29.95	DJ Software	Commodore 64/128	Strum-along-song disk comes with 15 songs, from "Hey Jude" to "Hard Day's Night."
Camus	$50	Conduit	Apple II; IBM PC	Set of exercises that train the ear to perceive musical notation.
Chord Power for Guitar	$39.95	Newarts	Commodore 64	Displays over 10,000 guitar chords with sound at user's request.
Chord Primer	$49.95	Dynacomp	IBM PC, PCjr	Program capabilities range from a built-in library of over 600 chords to a set of automated lessons on music theory for guitar.
Chords	$79	Wenger Computer Software	Apple II, II+, IIe	Intermediate or advanced music students drill and practice chord identification for ear training.
Christmas Classics	$9.95	Free Spirit Software	Commodore 64/128	"Joy to the World," "Deck the Halls," "Twelve Days of Christmas," and "Jingle Bells" are among the over 40 songs included.
Christmas, Volume 3	$15	Great Wave Software	Mac, Mac Plus	Collection of Christmas songs.
Classical Selection, Volume 5	$15	Great Wave Software	Mac, Mac Plus	Collection of favorite classical music.
Clef Notes	$39.95	Electronic Courseware	Apple II+, IIe, IIc; Commodore 64/128; IBM PC, PCjr; Tandy 1000	Drill-and-practice in identifying notes as they're placed on the treble, alto, tenor, and bass clefs.
Coco Notes	$12.95	CBS Interactive Learning	Atari eight-bit; Commodore 64	Players try to catch notes, create melodies, and fish for tunes. Teaches sound discrimination, musical patterns, and composition. For ages 6 and up.
Computer Song/Album/Music-Video Hits	$15.95	Sight & Sound Music Software	Commodore 64	Listen to hits of favorite artists while controlling computer-generated instrument sounds and special effects.
COMPUTE!'s Music System for the Commodore 128 & 64	$24.95	COMPUTE! Publications	Commodore 64/128	Enter, edit, and play the most sophisticated music possible on the Commodore 128 and 64 with *Enhanced Sidplayer*.
ConcertWare+ MIDI	$139.95	Great Wave Software	Mac, Mac Plus	Control any MIDI-compatible electronic keyboard, synthesizer, or drum machine. Record voices as you enter them monophonically on an electronic keyboard.
ConcertWare+, Version 3	$69.95	Great Wave Software	Mac, Mac Plus	Create, edit, print, and play music files; create new instrument sounds, as well. Some music and instruments included.
Deluxe Music Construction Set, Version 2.0	$99.95	Electronic Arts	Mac, Mac Plus; Amiga	Improved and redesigned to take full advantage of these powerful computers. Enter notes directly on the staff with the mouse or from onscreen keyboard.



Run to your dealer to check out this GREAT SOFTWARE

"...everything a good compiler should be...easy to use...efficient...offers a good range of optional features...excellent documentation...inexpensive."
Tom Benford, Commodore Magazine

Give your BASIC programs a boost!

Basic Compiler

Now anyone can speed up their BASIC programs by 3 to 35 times! Basic-64 and Basic-128 easily converts your programs into fast machine language or speedcode (takes up less space yet protects your programs from prying eyes) or a mixture of both. You can even compile programs written with extentions—Simon's Basic, VICTREE, BASIC 4.0, VideoBasic and others. When the compiler finds an error, it just doesn't stop, but continues to find any other errors as well. Supports overlays and has many other options. 128 version works in FAST mode and allows you to use all 128K of memory. If your program walks or crawls, give it the speed to *RUN!*

for **C-64 $39.95** for **C-128 $59.95**

Super C

C is one of today's most popular languages. It's easy to transport C source code from one computer to another. With ***Super C*** you can develop software or just learn C on your Commodore. ***Super C*** is easy to use and takes full advantage of this versatile language. Produces 6502 machine code and is many times faster than BASIC. Includes full-screen editor (search, replace and block operations), compiler, linker and handbook. Combine up to seven modules with the linker. Besides the standard I/O library, a graphic library (plot points, lines, fill) and a math library (sin, cos, tan, log, arctan, more) are included. Whether you want to learn C, or program in a serious C environment for your Commodore, ***Super C*** is the one to buy.

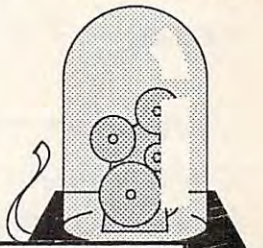
"...easy to use package with more power than most users should need...ultra-fast compile and link times make program development effortless." Adam Herst, Transactor

for **C-64 $59.95** for **C-128 $59.95**

PPM

Personal Portfolio Manager is the most comprehensive stock market portfolio management system available for the 64 or 128—For investors who need to manage stock portfolios, obtain up-to-the-minute quotes and news and perform selected analysis. Allows multiple portfolios for special intrests (high tech, low risk, income, etc.) and monitored individually. And the versatile report generator lets you produce any kind of report to analyze a portfolio or stock. You can even update your portfolio automatically using Dow Jones or Warner Computer Systems and your modem.

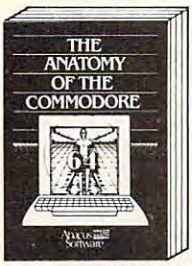
for **C-64 $39.95** for **C-128 $59.95**



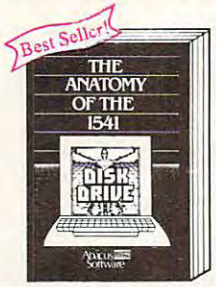
"...Personal Portfolio Manager will help you make the most of your money."
Jim Grubbs, RUN Magazine

"...a customized data base with advanced telecommunications features and a relatively sophisticated report generator. This combination is hard to beat on any microcomputer."
Ted Salamone, Commodore Magazine

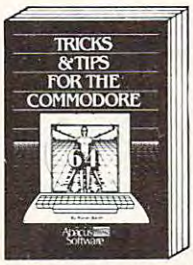
...and SUPER BOOKS!



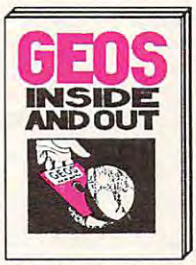
Anatomy of the C-64
Insider's guide to '64 internals. Graphics, sound, I/O, kernal, memory maps, and much more. Complete commented ROM listings. **300pp $19.95**



Anatomy of the 1541 Drive
Best handbook on this drive, explains all. Filled with many examples programs, utilities. Fully commented 1541 ROM listings. **500pp $19.95**

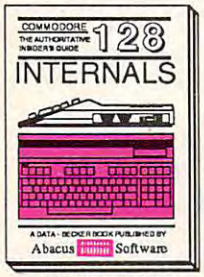


Tricks & Tips for the C-64
Collection of easy-to-use techniques: advanced graphics, improved data input, CP/M, enhanced BASIC, data handling and more. **275pp $19.95**

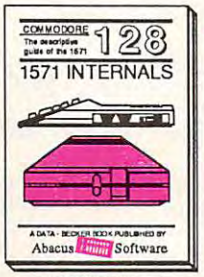


GEOS Inside and Out
Detailed info on GEOS. Add your own applications to GEOS. Edit icons. Constant display clock. Single-step through memory. **$19.95**

GEOS Tricks and Tips
Collection of helpful techniques for all GEOS users. Includes source for a font editor and a machine language monitor. **$19.95**



C-128 INTERNALS
Important C-128 information. Covers graphic chips, MMU, I/O, 80 column graphics and fully commented ROM listings, more. 500pp $19.95



1571 INTERNALS
Essential reference. Internal drive functions. Explains various disk and file formats. Fully-commented ROM listings. 450pp $19.95

C-128 TRICKS & TIPS
Fascinating and practical info on the C-128. 80-col hires graphics, bank switching. 300 pages of useful information for everyone. **$19.95**

C-128 PEEKS & POKES
Dozens of programming quick-hitters. techniques on the operating system, stacks, zero page, pointers, and BASIC. 240pp $16.95

C-128 BASIC 7.0 Internals
Get all the inside info on BASIC 7.0. This exhaustive handbook is complete with fully commented BASIC 7.0 ROM listings. **$24.95**

Call now for the name of your nearest dealer. Or order direct with your credit card by calling 616/241-5510. Add $4.00 per order for S&H. Foreign add $12.00 per item.

Other books and software also available. Call or write for your **free catalog.** Dealers inquires welcome–2000 nationwide.

Abacus
P.O. Box 7219
Dept. C5
Grand Rapids, MI 49510
Telex 709-101 • Fax 616/241-5021
Phone 616/241-5510

Product	Price	Publisher/ Vendor	System	Description
Ear Challenger Game	$39.95	Wenger Computer Software	Apple II, II+, IIe	An aural/visual game designed to increase tonal memory of a series of pitches. Seven levels of difficulty. For elementary, intermediate, and advanced students.
Early Music for Musicworks	$15	Recreations Software	Mac, Mac Plus	More than 60 music files of early music of many different styles, sounds, and moods.
Early Music Skills	$39.95	Electronic Courseware	Apple II+, IIe, IIc; Commodore 64/128; IBM PC; Tandy 1000	Note-recognition tutorial and drill for beginning music students. A MIDI version is available.
Early Music, Volume 4	$15	Great Wave Software	Mac, Mac Plus	Collection of music from the Renaissance period.
Ear Teacher	$79	Wenger Computer Software	Apple II, II+, IIe	Provides complete record keeping for students using the Intervals, Tunings, Chords, and Melodic games music programs.
Easy Guitar	$29.95	DJ Software	Commodore 64	Guitar instruction.
Elvis Classics	$29.95	DJ Software	Commodore 64/128	Strum-along-song disk with 15 songs. Includes "Teddy Bear," "Hound Dog," "Love Me Tender," and more.
Euphony Jr.	$19.95	TCO Software	Commodore 64/128	A collection of three hours of classical music for your listening enjoyment.
Euphony+	$29.95	TCO Software	Commodore 64/128	Create, edit, and play music in three voices. Print out your music scores.
FB01 Design	$139.95	Sonus	Commodore 64/128	A double-banked MIDI librarian and editor for use with the Yamaha FB01 FM sound generator.
Find That Tune	$39.95	Electronic Courseware	Apple II, II+, IIe, IIc, IIGS; Commodore 64/128	Aural/visual program with two difficulty levels.
GlassTracks	$69.95–$85	Sonus	Apple II+, IIe, Macintosh; Commodore 64; Atari 520/1040ST	A multifunctional MIDI recording studio.
Guitar Master 1.0	$49.95	Mastersoft	Commodore 64	A comprehensive program of instruction designed for students, amateurs, and professional guitarists.
Guitar Tutor, Volume 1	$49.95	Nappo Software	Mac, Mac Plus	Teaches beginning guitarists the correct finger positions of basic chords.
Guitar Wizard	$29.95–34.95	Baudville	Apple II+, IIe, IIc, IIGS, Mac; Atari eight-bit, ST; Commodore 64/128, Amiga; IBM PC, XT, AT	Learn and analyze scales, chords, and tunings for all types of fretted string instruments. Clear graphic displays of the fretboard, notes, intervals, and finger positions.
Halftime Battling Bands	$12.95	CBS Interactive Learning	Atari eight-bit; Commodore 64	Children choreograph and stage their own Be-Bop Bowl halftime show. Trip up the opposing band while creating your own marching tunes and band formations.
Hey Diddle Diddle	$20.95–$26.95	Spinnaker Software	Apple II, II+, IIe, IIc; Atari eight-bit; Commodore 64; IBM PC	Collection of 30 classic nursery rhymes featuring brilliant color graphics and lively music.
Imagination: Music	$34.95	Wiley Professional Software	Apple II, II+, IIe	Music is both heard and seen; each song can be repeated, edited further, or saved for future listening.
Incredible Musical Keyboard	$29.95	Sight & Sound Music Software	Commodore 64	Transforms the Commodore 64 into a musical instrument complete with black and white keys.
Instant Keyboard Fun I—MIDI	$39.95	Electronic Courseware	Apple II+, IIe; Commodore 64/128	Twenty-six songs the user plays on a synthesizer keyboard.
Instant Music	$49.95	Electronic Arts	Amiga	The Amiga accompanies you with the sound of three instruments while you create music using a mouse; a music composition and creativity program.
It's Only Rock'n'Roll	$29.95	Electronic Arts	Amiga	The first in a series of library disks for use with EA's *Deluxe Music Construction Set, Deluxe Video,* and *Instant Music.*
Kawasaki Rhythm Rocker	$26.95	Sight & Sound Music Software	Commodore 64	Combines color graphics with electronic instrument sounds and preprogrammed bass rhythms. Features a multitrack recording capability.
Kawasaki Synthesizer	$29.95	Sight & Sound Music Software	Commodore 64	Two-disk package that transforms the Commodore 64 into a programmable synthesizer and sound processor.
Keyboard Chords—MIDI	$79.95	Electronic Courseware	Apple II+, IIe; Commodore 64/128; IBM PC; Tandy 1000, 1200	Tutorial on major, minor, diminished, and augmented chords; a chord spelling drill; and a keyboard drill.
Learning Guitar Overnight	$39.95	Chipware	Commodore 64	Introduction to the joy of playing a guitar. Learn to strum songs in less than an hour.
Listen	$69	Imaja	Mac, Mac Plus	Interactive music program providing melodic and harmonic ear training.
Listen!	$39.95	Electronic Courseware	Apple II+, IIe, IIc; Commodore 64; IBM PC, PCjr; Tandy 1000, 1200	Three lessons designed to help increase the ability to perceive and identify intervals, basic chords, and seventh chords.
Magic Piano	$49.95	Edusoft	Apple II, II+, IIe, IIc	A music-learning system for any teacher who wishes to introduce music in the classroom. Transforms computer keys into piano keys.

Lyco Computer

Marketing & Consultants

Since 1981

Lyco Computer is one of the oldest and most established computer suppliers in America. Because we are dedicated to satisfying every customer, we have earned our reputation as the best in the business. And, our six years of experience in mail-order computer sales is your assurance of knowledgeable service and quality merchandise.

We fill 95% of all orders every month. Here's how: • lowest prices anywhere • multimillion $ factory fresh inventory • courteous, knowledgeable sales staff • 24-hour shipping on in-stock items.

Plus: • free shipping in U.S. on prepaid cash orders • no deposit on C.O.D. orders • no sales tax outside PA • full manufacturers' warranties apply • air freight, UPS Blue/Red shipping available.

Call Lyco Computer. See for yourself why so many customers keep coming back to Lyco for the best prices, the most complete inventory, and our fast and courteous service.

To order, call toll-free:
1-800-233-8760
In Penna.: 1-717-494-1030
Customer Service:
1-717-494-1670

Or write:
Lyco Computer, Inc.
P.O. Box 5088
Jersey Shore, PA 17740

MasterCard VISA

Risk-Free Policy: • prices show 4% cash discount; add 4% for credit cards • APO, FPO, international: add $5 plus 3% for priority mail • 4-week clearance required on personal checks • compatibility not guaranteed • return authorization required • we check for credit card theft

Price and availability subject to change without notice.

System Specials

Complete
COMMODORE 128 System

- Commodore 1902 A monitor
- Commodore 1571 Disk Drive
- Commodore 128 Computer

$759

(mouse available)

NOW AVAILABLE
COMMODORE PC 10-1 $749

COMMODORE PC 10-2 $899

Complete
COMMODORE 64C System

- Commodore 64 C Computer
- 1541 C Drive
- Seikosha SP-1000 VC Printer
- 2 joysticks

(90-day Warranty)
(reg. $789)

$509

Power and versatility at a great price!

BLUE CHIP
THE NEW PERSONAL COMPUTER

100% IBM PC/XT compatible -- GUARANTEED, or your money back!

(call for details)

COMMODORE HARDWARE

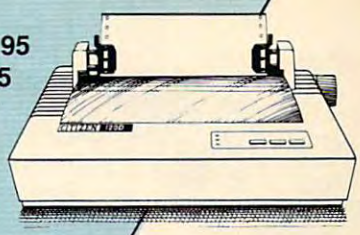
128 Computer	$275	C-1700 128K RAM	$109.95
1571 Disk Drive	$229	1750 RAM	$169.95
64C Computer	$175	Indus GT C-64 Drive	$179
1541 C Disk Drive	$185	GEOS	$CALL
1902 Monitor	$285	C-1351 Mouse	$39
1802C Monitor	$189	1670 Modem	$99

1-800-233-8760

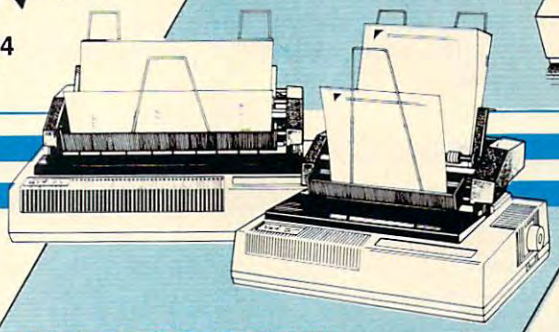
PRINTERS

Citizen 120-D . . .$179

Panasonic 1080i . .$195
Star NX-10 . . . . . $195
Star NP-10 . . . . . $169

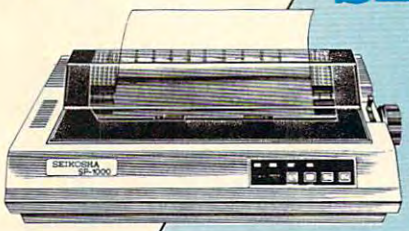


Seikosha SP1000VC . . $154
Seikosha SP1200Ai . . .$195



GREAT DEALS ON PRINTERS!

SEIKOSHA



SP 1000AP IIc $179
SP 1200AS R5232 . $195
SL-80Ai $375
MP 1300Ai $439
MP 5300Ai $549
BP 5420Ai $1075
1300 Color Kit $119
BP-5420 Ribbon . . $12.50
SP-1000Ribbon $8.50

LEGEND

808 . . . . . $159
1080 . . . . . $199
1380 . . . . . $229
1385 . . . . . $289

Toshiba

321 P/S $479
P341E $699
P351 Model II . . . . . $CALL

OKIDATA

Okimate 20 . . . . . $129
120 NLQ $209
292 w/interface $539
293 w/interface $679
182 . . . . . $235
192+ $365
193+ $559

star micronics inc

NX-10C $209
NL-10 . . . . . $229
NX-15 . . . . . .$329
NB-15 . . . . . .$889
SD-10 . . . . . .$259
ND-15 . . . . . .$425
SR-10 . . . . . .$469
NR-15 . . . . . .$529
NB24-15 . . . . . .$729

Panasonic

1091i $269
1092i $335
1592 . . . . . .$399
1595 . . . . . .$499
3131 . . . . . .$249
3151 . . . . . .$379
1080 AP IIc $239

JUKI

R5232 serial board $55
Juki 6300 . . . . . .$739
Juki 6100 . . . . . .$399

Monitors

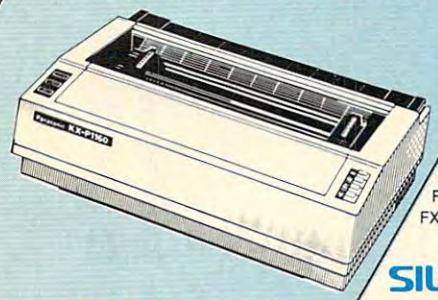
Teknika
MJ-305 . . . . . .$309
MJ-503 . . . . . .$529

Zenith
ZVM 1220 . . . . . .$89
ZVM1230 . . . . . .$89

Panasonic
TR-122 MYP 12" Am TTL . .$139
TR-122 M9P 12" Gr TTL . . .$139

Commodore
1902A Color $285
1802 C $189

NEC
Multisync $569



EPSON

LX 86 . . . . . .$209
EX 800 . . . . . $355
EX 1000 . . . . . .$579
LQ 800 . . . . . $449
LQ 1000 . . . . . .$659
FX 86E $369
FX 286E $519

SILVER REED

EXP 420P $209
EXP 600P $539
EXP 800P $649

CITIZEN

Premiere 35 . . . . . $469
MSP-10 . . . . . $285
MSP-15 . . . . . $385
MSP-20 . . . . . $325
MSP-25 . . . . . $485

DIABLO

D25 . . . . . .$519
635 . . . . . $799
D-80 IF $1029

The Avatex 1200hc

Modems

Avatex
1200 . . . . . $89
1200hc $119

SPECIAL PRICE

Atari
XM-301 . . . . . .$42.95
SX212 . . . . . .$CALL

Hayes
Smartmodem 300 . . . . . $125
Smartmodem 1200 . . . . . $369
Smartmodem 1200B $339
Smartmodem 2400 . . . . . $559
Micromodem IIe $125
Smart 300 Apple IIc $149

Commodore
1670 . . . . . $99

Zoom
PC 1200 ST (internal) $189
PC 1200 XL (internal) $259
PC 2400 ST (internal) $329
PC 2400 XL (internal) $349

We stock interfacing for Atari, Commodore, Apple and IBM

Vickie Blaker
Customer Service Department Manager

"Our Customer Service Department is one example of how much Lyco Computer cares about your satisfaction. Everyday we talk to people from all over the world, and our goal with each person who calls is to be as helpful, efficient, and courteous as possible. We're ready to find the answers -- from questions about the status of an order, to warranties, to product availability and price -- or guide you to someone who can. And we're **always** here to help you. . .before, during, and **after** your purchase."

RI ST SOFTWARE

. $24.95
$13.95
24.95
5
Graphic Library I, II, III . . . $15.75
arateka $18.75
t Shop $25.75
Shop Companion . . .$22.75
hop Paper $12.75
Kit $35.75
d $24.95
. . . . . $23.95
. . . . . $29.95
. . . . . $24.95
ent . . . $13.95
. . . . $13.95
. . .$14.95
$24.75
2.75
75
5
Sentry $24.95

li Tech Expressions:
art Ware $6.75
lay Paper $8.95
Vare $6.75
Vare $9.75

Vorld:
or 2 . . . . . .$15.95
. . . . . $22.95

ded:
. . . . . $26.95
. . . . . .$44.95
. . . . . $24.95
. . . $34.95
. $14.95

COMMODORE SOFTWARE

Broderbund:
Bank St. Writer 128K $32.75
Carmen San Diego $22.75
Champ. Lode Runner $24.75

Firebird:
Colossus IV $21.95
Elite $19.95
Frankie Goes to Italy $19.95
Pawn $28.95
Talking Teacher $28.95
Tracker $31.95
Starglider $24.95
Sentry $24.95

Unison World:
Art Gallery 2 . . . . . .$19.95
Print Master $25.95

Sublogic:
Baseball Stadium $15.95
Flight Simulator II $31.95
Jet Simulator $31.95
Night Mission Pinball $21.95
Scenery Disk $15.95
Sublogic Baseball $31.95
Sublogic Football $26.95

Epyx:
Destroyer $24.75
Fastload $24.75
Football $24.75
Karate Champ $19.75
Movie Monster $24.75
Multiplan $24.75
Sub Battle $23.95
Winter Games $23.95
Super Cycle $23.95
Graphics Scrapbook $14.95
St. Sports Basketball $23.95
World's Great Baseball . . .$19.95
Summer Games II $24.75
Vorpol Utility Kit $15.75
World Games $24.75

Firebird:
Colossus IV $21.95
Elite $19.95
Frankie Goes to Italy $19.95
Pawn $28.95
Talking Teacher $28.95
Tracker $31.95
Starglider $24.95
Sentry $24.95

Sublogic:
Flight Simulator II $32.95
Jet Simulator $26.95
Night Mission Pinball $22.95
Scenery #1-#6 . . . . . .$69.95

Epyx:
Destroyer $24.95
Karate Champ $19.95
Movie Monster $24.95
Winter Games $24.95
World Games $24.95
Wrestling $24.95

Firebird:
Colossus IV Chess $25.95
Elite $22.95
The Pawn $26.95
Starglider $28.95

Unison World:
Art Gallery 2 . . . . . .$19.95
Print Master $25.95

Sublogic:
Baseball Stadium $15.95
Flight Simulator II $31.95
Jet Simulator $31.95
Night Mission Pinball $21.95
Scenery Disk $15.95
Sublogic Baseball $31.95
Sublogic Football $26.95

Epyx:
Destroyer $24.75
Fastload $24.75
Football $24.75
Karate Champ $19.75
Movie Monster $24.75
Multiplan $24.75
Sub Battle $23.95
Winter Games $23.95
Super Cycle $23.95
Graphics Scrapbook $14.95
St. Sports Basketball $23.95
World's Great Baseball . . .$19.95
Summer Games II $24.75
Vorpol Utility Kit $15.75
World Games $24.75

Strategic Simulations:
Battlegruppe $38.95
Colonial Conquest $25.95
Gettysburg $35.95
Phantasie II $25.95
Shard of Spring $25.95
Wizards Crown $25.95

Sublogic:
Flight Simulator II $32.95
Jet Simulator $26.95
Night Mission Pinball $22.95
Scenery #1-#6 . . . . . .$69.95

Epyx:
Destroyer $24.95
Karate Champ $19.95
Movie Monster $24.95
Winter Games $24.95
World Games $24.95
Wrestling $24.95

Firebird:
Colossus IV Chess $25.95
Elite $22.95
The Pawn $26.95
Starglider $28.95

Microprose:
Conflict in Vietnam $24.95
Crusade in Europe $24.95
Decision in Desert $24.95
F-15 Strike Eagle $22.95
Silent Service $22.95

Microleague:
Micro Leag. Baseball $25.95
General Manager $25.95
Stat Disk $17.95
'86 Team Disk $14.95

AMIGA SOFTWARE

ard $24.95
1 . . . . . .$14.95

Epyx:
Apshai Trilogy $24.75
Destroyer $24.75
Karate Champ $19.75
Movie Monster $24.75

Leather Goddesses $25.75
Term Paper $35.75
Champ. Basketball $26.95
Championship Golf . . . . . .$New

Microprose:
Crusade in Europe $24.95
Decision in Desert $24.95
F-15 Strike Eagle $22.95
Silent Service $22.95

Microleague:
Micro Leag. Baseball $25.95
General Manager $25.95
Stat Disk $17.95
'86 Team Disk $14.95

Broderbund:
Airheart $22.75
Ancient Art of War $25.75
Print Shop $29.75
Print Shop Companion . . $25.75
Carmen S. Diego (USA) . . $28.75
On Balance $62.75
Bank St. Writer + ...$44.75

Strategic Simulations:
Battlegruppe $38.95
Colonial Conquest $25.95
Gettysburg $35.95
Phantasie II $25.95
Shard of Spring $25.95
Wizards Crown $25.95

Sublogic:
Flight Simulator II $32.95
Jet Simulator $26.95
Night Mission Pinball $22.95
Scenery #1-#6 . . . . . .$69.95

Epyx:
Destroyer $24.95
Karate Champ $19.95
Movie Monster $24.95
Winter Games $24.95
World Games $24.95
Wrestling $24.95

Activision:
Pebble Beach $32.95
Portal $28.95
Champ. Basketball $26.95
Ballyhoo $25.75
Enchanter Trilogy $49.75
Leather Goddesses $25.75
Moonmist $25.75
Music Studio $32.75

Microprose:
Conflict in Vietnam $24.95
Crusade in Europe $24.95
Decision in Desert $24.95
F-15 Strike Eagle $22.95
Silent Service $22.95

Microleague:
Micro Leag. Baseball $25.95
General Manager $25.95
Stat Disk $17.95
'86 Team Disk $14.95

Epyx:
Apshai Trilogy $24.75
Destroyer $24.75
Karate Champ $19.75
Movie Monster $24.75

IBM SOFTWARE

Activision:
Pebble Beach $32.95
Portal $28.95
Champ. Basketball $26.95
Ballyhoo $25.75
Enchanter Trilogy $49.75
Leather Goddesses $25.75
Moonmist $25.75
Music Studio $32.75

Unison World:
Art Gallery 2 . . . . . .$18.95
News Master $55.75
Print Master $36.75

Broderbund:
Ancient Art of War $28.95
Print Shop $37.75
Print Shop Companion $31.75
Graphic Library I or II $21.75
Karateka $21.75
Toy Shop $39.95
Bank St. Writer + ...$54.95

Unison World:
Art Gallery 2 . . . . . .$18.95
News Master $55.75
Print Master $36.75

Diskettes

5-1/4

Maxell:
SSDD $9.99
DSDD $12.99

Activision:
Borrowed Time $26.95
Champ. Basketball $26.95
Championship Golf $32.95
Hacker $26.95
Hacker 2 . . . . . .$29.95
Little People $29.95
Mindshadow $26.95
Music Studio. $35.95
Tass Times $26.95

Strategic Simulations:
Computer Baseball $24.95
Kampfgruppe $34.95

Sublogic:
Flight Simulator II $32.95
Jet Simulator . . . . . .$CALL

Epyx:
Apshai Trilogy $22.95
Rogue $22.95
Winter Games $22.95

Firebird:
Pawn $26.95
Starglider $26.95

Unison World:
Print Master $24.95
Art Gallery 1 or 2 . . . . . .$18.95

VIP:
VIP Professional $115

Microprose:
Silent Service $24.95

IBM SOFTWARE

Rogue $19.75
Winter Games $24.75
World Games $24.75

Strategic Simulations:
Road War 2000 . . . . . .$24.95
Kampfgruppe $36.95
Shard of Spring $24.95
Battle of Antetiem $38.75
Computer Baseball $14.95
Gettysburg $38.75
Knights in Desert $25.75

Sublogic:
Jet Simulator $32.95
Scenery Japan $15.95
Scenery San Fran. $15.95
Scenery #1-#6 . . . . . .$69.95

Broderbund:
Ancient Art of War $28.95
Print Shop $37.75
Print Shop Companion $31.75
Graphic Library I or II $21.75
Karateka $21.75
Toy Shop $39.95
Bank St. Writer + ...$54.95

Unison World:
Art Gallery 2 . . . . . .$18.95
News Master $55.75
Print Master $36.75

Bonus:
SSDD $6.99
DSDD $7.50

SKC:
SSDD $8.50
DSDD $9.50
DSHD $18.95

Verbatim:
SSDD 9.99
DSDD $12.99

3.5

3M:
SSDD $16.99
DSDD $23.99

Maxell:
SSDD $15.99
DSDD $21.99

Verbatim:
SSDD $16.99
DSDD $24.99

SKC:
SSDD $14.99
DSDD $19.99

We stock over 5000 titles of software. If it's not listed, call for price and availability!

130XE System $429

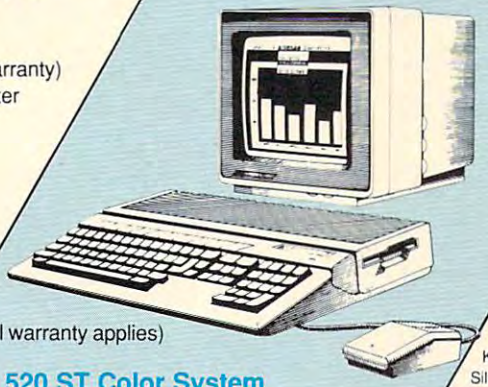
(full manufacturer's warranty)
- Atari 130 XE Computer
- Star NX-10 Printer
- 1050 Drive

Purchase orders accepted from educational institutions. Also, ask about volume discounts!

$899

ATARI

Atari 1040 Mono System with Seikosha SP1200Ai Printer

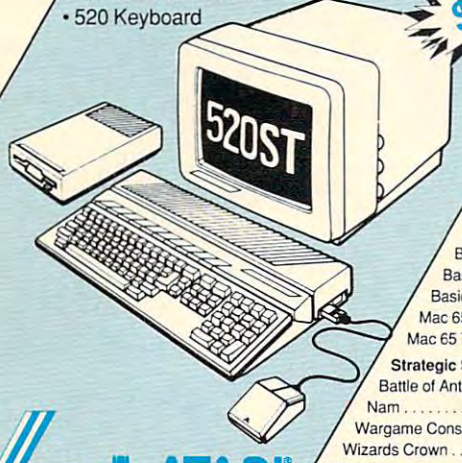


(full warranty applies)

Atari 520 ST Color System

(full manufacturer's warranty)
- SC-1224 Color Monitor
- SF-354 Disk Drive
- 520 Keyboard

$754



ATARI HARDWARE

SF 314 Disk Drive	$219.95
SF 354 Disk Drive	$175.95
1050 Drive (XE, XL)	$139.95
SHD 204 20 MEG Dr.	$589
65XE	$99
520 ST Mono	$515
Indus GT Atari Dr.	$179
130XE Computer	$139
65XE Computer	$99

ATARI SOFTWARE

Access:	
Leader Board	$24.95
Tournament #1	$14.95
Triple Pack	$14.95
10th Frame	$24.95
Activision:	
Hacker	$15.95
Hitch Hikers	$22.95
Leather Goddesses	$22.95
Moonmist	$22.95
Music Studio	$22.95
Microprose:	
Conflict in Vietnam	$24.95
Decision in Desert	$24.95
F-15 Strike Eagle	$22.95
Kennedy Approach	$19.95
Silent Service	$22.95
Top Gunner	$19.95
Microleague:	
Micro League Baseball	$24.95
General Manager	$24.95
Stat Disk	$17.95
'86 Team Disk	$14.95
Broderbund:	
Print Shop	$25.75
Print Shop Companion	$22.75
Graphic Library I, II, III	$15.75
Karateka	$18.75
Bank St. Writer	$29.75
Optimized Systems:	
Action	$46.95
Action Tool Kit	$18.95
Basic XE	$46.95
Basic XL	$36.95
Basic XL Tool Kit	$18.95
Mac 65	$46.95
Mac 65 Tool Kit	$18.95
Strategic Simulations:	
Battle of Antetiem	$32.95
Nam	$24.95
Wargame Construction	$18.95
Wizards Crown	$24.95
Warship	$39.95
Sublogic:	
Flight Simulator II	$31.95
Night Mission Pinball	$21.95
Scenery Japan	$15.95
Scenery #1 - #6	$69.95
Scenery Disks, each	$14.95
Epyx:	
Eidolon	$22.95
Karate Champ	$18.95
Koronis Rift	$22.95
Summer Games	$24.95
Epyx:	
Apshai Trilogy	$24.95
Super Cycle	$24.95
World Games	$24.95
Wrestling	$24.95
Karate Champ	$24.95
Firebird:	
Pawn	$26.95
Starglider	$26.95
Golden Path	$28.95
Tracker	$28.95
VIP:	
Accounts	$174.95
VIP Professional	$124.95
VIP Professional Lite	$74.95
Unison World:	
Art Gallery 1 or 2	$18.95
Print Master	$24.95
Zoom:	
Zoomracks	$54.95
Zoomracks II	$79.95
Data Soft:	
Mercenary	$26.75
Mercenary - 2nd City	$17.95
Alt. Reality - The City	$34.95
Alt. Rlty. - The Dungeon	$26.75
Microprose:	
Silent Service	$24.95
F-15 Strike Eagle	$24.95
Optimized Systems:	
Personal Pascal	$48.95
Personal Prolog	$51.75
Strategic Simulations:	
Phantasie	$24.95
Phantasie II	$24.95
Road War 2000	$26.95
Colonial Conquest	$24.95
Epyx:	
Apshai Trilogy	$24.95
Super Cycle	$24.95
World Games	$24.95
Wrestling	$24.95
Karate Champ	$24.95
Firebird:	
Pawn	$26.95
Starglider	$26.95
Golden Path	$28.95
Tracker	$28.95
VIP:	
Accounts	$174.95
VIP Professional	$124.95
VIP Professional Lite	$74.95
Unison World:	
Art Gallery 1 or 2	$18.95
Print Master	$24.95
Zoom:	
Zoomracks	$54.95
Zoomracks II	$79.95

apple SOFTWARE

Access:	
Triple Pack	$14.95
Activision:	
Enchanter Trilogy	$49.75
Game Maker	$31.75

Product	Price	Publisher/ Vendor	System	Description
Making Music on Micros	$69.95	Random House Software	Apple II, II+	Learn BASIC programming and music theory at the same time.
Master Composer	$39.95	Access Software	Commodore 64	Takes full advantage of the sound chip to produce all types of music, from simple melodies to intricate compositions.
Melodian ConcertMaster	$59.95	Melodian Systems	Commodore 64/128	Combines the capabilities of a music synthesizer, recording studio, and video display. *ConcertMaster* creates an environment for experimenting and learning about music.
Melodian Keyboard with ConcertMaster	$159	Melodian Systems	Commodore 64/128	This unique system is a major advance in the teaching, learning, and enjoyment of music.
Melodian RhythmMaster	$59.95	Melodian Systems	Commodore 64/128	*RhythmMaster* helps develop perfect timing through the use of color-coded video-displayed notes.
Melodian ScoreMaster	$59.95	Melodian Systems	Commodore 64/128	Program your music and print it out in music notation which other musicians can read and play. Any music recorded with *ConcertMaster* can be printed.
Melodic Games	$79	Wenger Computer Software	Apple II, II+, IIe	Drill-and-practice memory dictation for ear training.
MIDI/4+	$129.95	Passport Designs	Apple II, II+, IIe; Commodore 64	Four-channel multitrack composing and recording tool for MIDI.
MIDI/8+	$169.95	Passport Designs	Apple IIe; Commodore 64/128	Eight-channel multitrack recording software that turns your computer and MIDI keyboard into a professional recording studio.
MIDI Jazz Improvisation	$79.00	Electronic Courseware	Apple II+, IIe, IIc	Provides instrumental and vocal students with play-along material to learn jazz improvisation using original tunes.
Midimac: Patch Librarian—Casio CZ	$75	Opcode Systems	Mac, Mac Plus	Made for the Casio CZ synthesizer with several banks of patches to store thousands of sounds with MIDI capabilities.
MIDI Processor	$149.95	Sonus	Commodore 64/128	A processing program for use with the *Super Sequencer* or *Studio I* sequencers.
MIDI Recording Studio	$39	Dr. T's Music Software	Atari ST	A MIDI recording program for those just beginning to work with MIDI, with a stripped-down version of parts of the Dr. T *Keyboard Controlled Sequencer*.
MIDIsoft Studio	$99	Passport Designs	Atari ST	A multitrack recording studio that works with the ST and any MIDI-equipped instrument.
MIDI Tech 64	$99.95	Sonus	Commodore 64/128	A full-featured monitor/system-exclusive librarian program.
MIDI Voice Librarian	$69.95	Passport Designs	Apple II+, IIe, IIc; Commodore 64/128	Over 100 great new sounds for MIDI keyboard. Load up to four banks of 32 sounds at any moment.
Musical Computer I and II, Version 1.0	$34.95	Computer Applications Tomorrow	Apple II+; Atari eight-bit; Commodore 64	Teaches music fundamentals. Covers note reading, sharps and flats, tempo definitions, and more.
Music Box I	$59.00	Wenger Computer Software	Apple II, II+, IIe, IIc; Commodore 64	Four programs designed to aid students in learning and remembering music symbols.
Music by Matrix	$29.95	Dynacomp	Commodore 64	Audiovisual aid to help the student understand chords and scales in terms of the intervals involved.
The Music Class	$39–49 each	Wenger Music Software	Apple II, II+, IIe, IIc, IIGS	A five-part music-instruction series, including Fundamentals, Rhythm, Ear Training, Music Symbols, and Note Reading.
Music Concepts, Version 1.0	$59.95	Ventura Educational Systems	Apple II, II+, IIe, IIc	Introduce the concepts of music theory, the history of music as we know it, and even the science of sound.
Music Construction Set	$34.95–$69.95	Electronic Arts	Apple II, II+, IIe, IIc, IIGS; Atari eight-bit; Commodore 64/128; IBM PC, PCjr, PC XT	A computer music program that everyone can enjoy. Doesn't require years of piano lessons or learning computer codes.
Music Editor	$20	Affordable Software	IBM PC	Compose songs with as many as 500 notes per song.
Music Logo	$99.95	Terrapin	Apple II, II+, IIe, IIc	Explores musical structure and extends the user's musical understanding and appreciation.
Music Made Easy	$29.95	Alfred Publishing	Apple II+, IIe; Commodore 64	Teaches the basics of music in a step-by-step course. Lessons are reinforced with drills and quizzes.
Music Magic	$30	Dayline Software	Commodore 64/128	Play your favorite songs and/or compose your own music.
Musicman	$29.95	Zephyr Services	Apple II, II+, IIe; IBM PC, XT, PCjr	Try your hand at composing music right on the screen with standard musical methods. Save compositions on disk or play some of the sample music provided.
Music of the Masters: I, II, III, and IV	$9.95	Free Spirit Software	Commodore 64/128	Collections of works by major classical composers. Instrument simulations include violin, piano, harpsichord, flute, and guitar.
Music of the Masters V	$9.95	Free Spirit Software	Commodore 64/128	Approximately one hour of popular themes from the best-known classical works, using various instrument simulations.
Music Processor	$24.95	Sight & Sound Music Software	Commodore 64	Create, edit, play, and compose your own musical arrangements.
Music Program	$19.95	Micro Demon	TRS-80 Model 100	Turns any Model 100 into a musical instrument by modifying the sound routine.
Music: Rhythm	$29	MECC	Commodore 64/128	Stimulating practice on rhythmic fundamentals. For beginning- to advanced-level music students.

Product	Price	Publisher/ Vendor	System	Description
Music: Rhythm and Pitch	$29	MECC	Atari eight-bit; Commodore 64	Three disks which can be used singly or in a combination to provide practice at successive levels of difficulty.
Music: Scales and Chords	$29	MECC	Atari eight-bit; Commodore 64/128	Music theory and drill-and-practice.
Music Shop	$149.95	Passport Designs	Commodore 64/128	Compose, edit, print, and play back with a joystick, easy-to-use pull-down menus, and your MIDI keyboard.
Music Studio	$34.95–$59.95	Activision	Atari eight-bit, ST; Commodore 64/128; Amiga; IBM PCjr; Tandy 1000; Apple IIGS	Music, lyrical composition, and audio synthesis program that lets you orchestrate, mix, create sounds, and even invent new sounds.
Music System	$39.95	Firebird	Commodore 64/128	A multitracking sound system. Use your Commodore keyboard to enter and correct music with the cassette-recorder-style multitracking functions.
MusicWorks	$49.95	Hayden Software	Mac	Provides all the tools needed for anyone to create and edit music, from simple melodies to fully orchestrated symphonies. Music can be composed on a standard musical staff or on a player-piano grid.
Notable Phantom	$19.95	Designware	Apple II+, IIe, IIc; Commodore 64; IBM PC, PCjr	Learn to play a keyboard instrument and to read music, with the help of funny ghosts, spiders, and The Notable Phantom.
Notes	$19.95	Comput-Ability	Apple II+, IIe, IIc	Develop speed and accuracy in identifying each musical note by its letter name. For ages 6 and up.
The Orchestrator	$49.95	Intersect Software	Atari ST	A music composition and entertainment system for both the experienced and beginning musician. MIDI compatible.
Party Songs	$15.95	John Henry Software	Commodore 64/128	A sing-along software program with old-time favorites.
Patch Librarian—Yamaha DX21/27/100	$75	Opcode Systems	Macintosh, Macintosh Plus	Use Mac disks to store thousands of sound patches for the Yamaha DX synthesizer. Takes the place of expensive RAM cartridges. Makes using inconvenient cassette-tape storage of sounds obsolete.
Personal Musician	$29.95	Creative Software/ Activision	IBM PC, PCjr	Experiment with computer-generated musical tones as you learn to read music and write your own original songs.
Player Piano	$19.95	Dynacomp	Atari	Turn your Atari into a player piano.
Rock 'N' Rhythm	$26.95	Spinnaker Software	Atari eight-bit; Commodore 64/128	Expand and develop your music skills by taking charge of your own recording studio.
RX Librarian	$49.95	Sonus	Commodore 64/128	A MIDI librarian that works with the Yamaha RX11 and RX21 drum machines.
Song Painter	$59.95	Rubicon Publishing	Mac, Mac Plus, Mac XL	Turns the Mac into a four-voice synthesizer that lets you create your own music with no knowledge of musical notation.
Songwriter	$19.95	Mindscape	Apple II+, IIe, IIc; Atari eight-bit; IBM PC, XT, PCjr	Colorful graphics combined with editing functions for over 28 different songs. Connector cable is included to hook up to stereo.
Sound Development System	$29.95	Dynacomp	Commodore 64	Create and place sound effects and music within your own BASIC or machine language programs.
Soundscape Pro MIDI Studio	$149	Mimetics	Amiga	A MIDI recording studio consisting of several interrelated MIDI modules.
Sound Tracks	$49	MECC	Apple II, II+, IIe, IIc	Turn your computer into a musical keyboard with this package. For ages 5–12.
Staff Master	$45	Micro Learningware	Apple II, IIe, IIc	Three programs for the beginning music student. Excellent graphics. For grade-level 4 and up.
Stickybear Music, Version 1.0	$39.95	Weekly Reader Family Software	Apple II, II+, IIe, IIc, IIGS	Compose a piece of music, play it, change the tempo, or go back and change notes or sections.
Strum-Along Songs	$69.95	DJ Software	Commodore 64/128	Play and sing your favorite songs on your guitar or keyboard accompanied by your own backup band. Each disk includes 15 easy-to-play songs.
Studiomac, Version 1.3	$125	Creative Solutions	Mac	Create music and play it out over a Casio CZ101 synthesizer.
SYNTHY-64	$17.95	Abacus Software	Commodore 64	A music and sound synthesizer that can duplicate a piano, banjo, flute, drum, or almost any other instrument. You can also make special-effects sounds in an endless variety of combinations.
Terpsichore	$49.95	Great Wave Software	Macintosh, Macintosh Plus	Music for exclusive use with ConcertWare+ and ConcertWare+ MIDI on the Macintosh.
3001 Sound Odyssey	$26.95	Sight & Sound Music Software	Commodore 64	An educational odyssey that explores the basics of electronic music synthesis and the construction of sound effects.
12-Bar Tunesmith	$39.95	Electronic Courseware	Apple II+, IIe, IIc; Commodore 64/128; IBM PC, PCjr; Tandy 1000, 1200, 3000	Helps the young compose and play simple melodies using bar-graph notation. Choose from four different pitch durations and hear tunes played at varying tempos.
Xylophone/Square Puzzle	$8.95	Kidware	Commodore 64/128, TI	Play any of nine songs or program your own.

The data for this guide was supplied by MENU—The International Database Corporation. For further evaluative information, or to insure that your product is included in the database, contact MENU, 1520 South College Avenue, Fort Collins, Colorado 80524. The toll-free number is 1-800-THE-MENU.

Rememory

Charles Harbert

How good is your memory? This program lets you test your memory against the computer or a friend. The original version is written for the Commodore 64. We have added new translations for the Amiga, IBM PC/PCjr, Apple II series, and Atari 400, 800, XL, and XE. The Commodore and Atari versions require at least one joystick. The IBM PC/PCjr version requires BASICA and a color/graphics card for the PC and Cartridge BASIC for the PCjr. The Apple II version works on any Apple II-series computer, under DOS 3.3 or ProDOS.

"Rememory" is a game that will push your powers of concentration and memorization to the limit. Type in the program listed for your computer and save it. Read the general instructions and refer to the specific notes for your computer before you begin to play.

Playing Rememory

Rememory is played on a grid containing 54 boxes arranged in a 9 × 7 matrix. Each box contains a graphics shape, and there are many matching shapes within the grid. The object of the game is to find all of the matches in the playing grid by selecting any two boxes at a time.

The graphics cursor (mouse pointer in the Amiga version) indicates your current postion on the game screen. Move the cursor to the box you wish to select using the joystick, mouse, or cursor controls, depending on which computer you are using. When you select the box, the computer displays the shape which it contains.

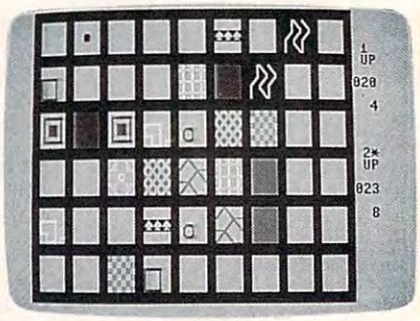
A turn consists of two selections. After you select both boxes, the computer displays both of them briefly. If the two shapes you selected are identical, you have scored one match, and those shapes remain visible on the board. If the shapes do not match, the computer erases them, and it is your job to remember where those shapes were found. The computer scrambles the shapes at the beginning of each game, so you won't know where a given shape is found until you uncover it.

Rememory can be played with one or two players. When you play alone, the object is to match all the shapes in the fewest number of turns. For a two-player game, the goal is to score more matches than your opponent. You get an extra turn every time you succeed in making a match. If you set a time limit for each move (for instance, 20 or 30 seconds), Rememory can be a fast-paced, exciting two-player game.

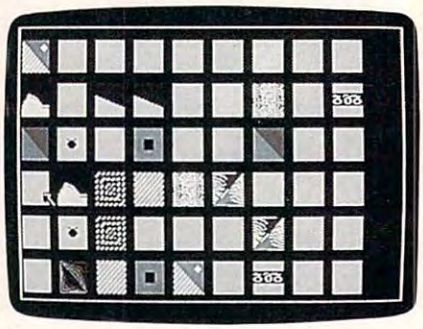
When you run the program, it asks how many players will play the game. Enter the number of players, 1 or 2. Then the program asks how many matches will be required to finish the game. If you enter the maximum number, 27, you will have to match every pair of shapes in the grid to finish. If you choose a lower number, the game ends when you achieve the designated number of matches. The right side of the screen displays the current score.

Commodore 64 Version

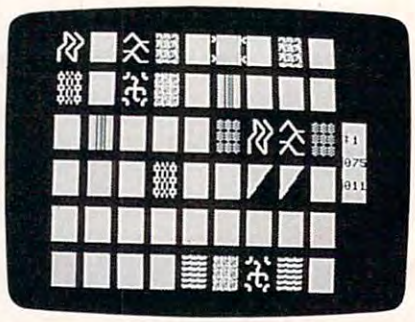
This version of Rememory (Program 1) can be played with one or two joysticks. If you are using only one joystick, plug it into port 2.



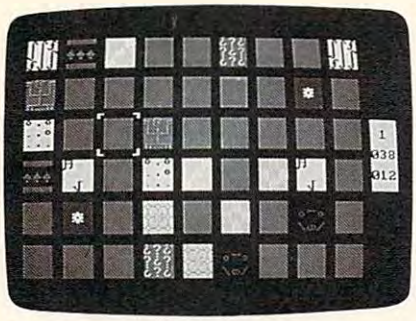
"Rememory" for the Commodore 64, a challenging memory game. This version uses custom machine language subroutines to speed up its graphics.



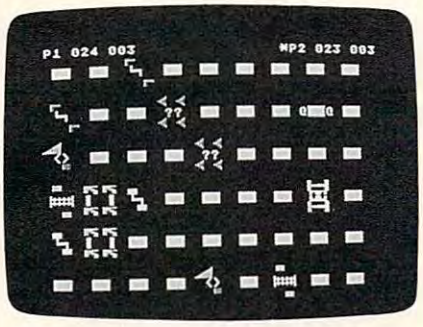
The Amiga version of "Rememory" uses a 32-color palette and color cycling to enhance the game's visual appeal and difficulty.



The Apple II version of "Rememory" is played with the keyboard and runs with either DOS 3.3 or ProDOS.



"Rememory" for the IBM PC/PCjr.



A custom display list is used to achieve the graphics effects in "Rememory" for the Atari 400, 800, XL, and XE.

Amiga Version

The Amiga game (Program 2) is played with the mouse. Move the mouse pointer to the desired box and press the left mouse button to select it. In the two-player game, the colors of the window border change to indicate whose turn it is. To add to the interest and difficulty, this program uses color cycling to change the colors of the graphics shapes.

IBM PC/PCjr Version

This version of Rememory (Program 3) requires BASICA and a color/graphics card for the PC, and it requires Cartridge BASIC for the PCjr. Move the cursor with the cursor keys and press Enter to select a box. For a two-player game, the scores are displayed on a red or green background. The cursor changes color to indicate whose turn it is.

Apple II Version

Rememory for the Apple II (Program 4) runs on any Apple II–series computer under either DOS 3.3 or ProDOS. This program is played with the keyboard. Press the I, J, K, and M keys to move the cursor up, left, right, and down, respectively. Press the space bar to select a box. In the two player game, the asterisk (*) indicates whose turn it is.

Atari 400, 800, XL, And XE Version

The Atari version of Rememory (Program 5) can be played with one or two joysticks. If you use only one joystick, it should be plugged into port 1.

For instructions on entering these programs, please refer to "COMPUTE!'s Guide to Typing In Programs" elsewhere in this issue

Program 1: Commodore 64 Rememory

```
AP 10 RO=-2:CO=2
BD 20 DIM SYM(27,12):DIM BT(25
      ,40):DIM MAT(53,16)
JG 30 POKE 252,0:POKE 253,0:RE
      STORE
KC 40 POKE 53280,14:POKE 53281
      ,14
BQ 50 MA=0:S(0)=1000:S(1)=1000
      :C(0)=0:C(1)=0
FH 60 GOSUB1290:GOSUB450:GOSUB
      750:GOSUB1080:GOSUB1840:
      GOSUB610:GOTO320
BH 70 IFMA=NMTHEN GOTO140
AX 80 GOSUB1580:LET B1=BX:GOSU
      B1500
DP 90 GOSUB1580:LET B2=BX:IF B
      1=B2 THEN GOTO90
SJ 100 GOSUB1500
KJ 110 IF MAT(B1,0)=MAT(B2,0)
      {SPACE}THEN GOTO210
HK 120 GOSUB1410:LET BX=B1:GOS
      UB1410:LET BX=B2
MQ 130 GOTO260
KH 140 PRINT "{HOME}{RVS}{WHT}
      {8 DOWN}{6 RIGHT}PRESS
      {SPACE}ANY KEY TO CONTI
      NUE"
FB 150 FOR A=0TO10:GET B$:NEXT
CE 160 GET A$:IF A$=""THENGOTO
      160
PC 170 INPUT "{2 DOWN}{RVS}
      {9 RIGHT}PLAY AGAIN(Y O
      R N)";Z$
BP 180 IF MID$(Z$,1,1)="N" THE
      N GOTO200
RA 190 GOTO30
MB 200 END
GP 210 LET MAT(B1,1)=99:LET MA
      T(B2,1)=99
CF 220 LET C(UP)=C(UP)+1:LET M
      A=MA+1
MP 230 LET S(UP)=S(UP)+1
AX 240 IF PL=2THENGOTO310
DE 250 GOTO270
GK 260 LET S(UP)=S(UP)+1
FR 270 IF PL=1 THEN GOTO310
ED 280 IF UP = 0 THEN GOTO300
CM 290 LET UP=0:GOTO310
RP 300 LET UP=1
SG 310 POKE 1024+40*3+39,96:PO
      KE1024+40*12+39,96
FF 320 IF UP=0 THEN POKE1024+4
      0*3+39,42
PA 330 IF UP=1THEN POKE 1024+4
      0*12+39,42
PA 340 POKE 251,UP:IF JS=1 THE
      N POKE 251,0
PP 350 PRINT"{HOME}{3 DOWN}
      {38 RIGHT}1"
QB 360 PRINT"{38 RIGHT}UP":LET
       S$=STR$(S(0))
GP 370 PRINT"{37 RIGHT}";RIGHT
      $(S$,3)
AM 380 LET C$=STR$(C(0))
CE 385 PRINT"{38 RIGHT}";RIGHT
      $(C$,2)
AS 390 IF PL=1 THEN GOTO70
GA 400 PRINT"{HOME}{12 DOWN}
      {38 RIGHT}2"
MH 410 PRINT"{38 RIGHT}UP"
DX 420 LET S$=STR$(S(1)):PRINT
      "{37 RIGHT}";RIGHT$(S$,
      3)
SS 430 LET C$=STR$(C(1))
PP 435 PRINT"{38 RIGHT}";RIGHT
      $(C$,2)
KB 440 GOTO70
GS 450 FOR R=0TO 24
GX 460 FOR C=0TO39
XS 470 LET BT(R,C)=88
FB 480 NEXT C
BK 490 NEXT R
BF 500 LET T=00
CK 510 FOR R=0TO 20 STEP 4
KM 520 FOR C=0TO32STEP4
GS 530 BT(R+1,C+1)=T:BT(R+1,C+
      2)=T
EQ 540 BT(R+1,C+3)=T:BT(R+2,C+
      1)=T
GJ 550 BT(R+2,C+2)=T:BT(R+2,C+
      3)=T
MK 560 BT(R+3,C+1)=T:BT(R+3,C+
      2)=T
JH 570 BT(R+3,C+3)=T:T=T+1
CH 580 NEXT C
XA 590 NEXT R
HP 600 RETURN
HE 610 REM DRAW BOARD
QA 620 PRINT "{CLR}":C=0:X=224
AJ 630 FOR V=0TO24STEP4:GOSUB6
      60:NEXT
JS 640 FOR H=0TO36STEP4:GOSUB7
      10:NEXT
PX 650 RETURN
RM 660 FOR H=0TO36
FS 670 POKE 1024+40*V+H,X
MA 680 POKE 55296+40*V+H,C
MB 690 NEXT H
RA 700 RETURN
DA 710 FOR V=0TO24
DK 720 POKE1024+40*V+H,X:POKE5
      5296+40*V+H,C
KM 730 NEXT V
RF 740 RETURN
RR 750 REM INIT SYM TABLE
HG 760 FORX=0TO26
KF 770 FOR Y=0TO11
DC 780 READ SYM(X,Y):NEXT Y
BX 790 NEXT X
CK 800 RETURN
FD 810 DATA 0,0,0,96,96,96,193
      ,193,193,96,96,96
SF 820 DATA 0,2,0,96,96,96,96,
      90,96,96,96,96
QE 830 DATA 0,2,0,120,120,120,
      120,120,120,120,120,120
CX 840 DATA 0,2,0,96,224,96,22
      4,224,224,96,224,96
PA 850 DATA 0,2,0,230,230,230,
      230,230,230,230,230,230
RQ 860 DATA 0,15,0,233,96,223,
      96,87,96,95,96,105
AM 870 DATA 0,9,0,224,223,96,9
      5,224,223,96,95,224
ME 880 DATA 0,0,0,231,205,205,
      206,206,206,229,205,229
JH 890 DATA 0,1,0,85,67,73,74,
      67,73,74,67,75
KG 900 DATA 0,6,0,214,214,214,
      214,214,214,214,214,214
GA 910 DATA 0,3,0,96,96,96,79,
      80,96,80,122,112
MP 920 DATA 0,11,0,127,127,127
      ,127,127,127,127,127,12
      7
DH 930 DATA 0,1,0,224,224,224,
      224,224,224,224,224,224
```

```
XJ 940 DATA 0,9,0,112,67,110,9
       3,224,93,109,67,125
AF 950 DATA 0,6,0,78,77,100,10
       0,78,77,78,77,100
PG 960 DATA 0,6,0,85,114,73,10
       7,86,115,74,113,75
KC 970 DATA 0,2,0,224,224,224,
       224,224,224,224,224,224
RB 980 DATA 0,9,0,110,0,112,10
       9,91,125,85,113,75
RB 990 DATA 0,4,0,224,224,224,
       224,224,224,224,224,224
SG 1000 DATA 0,7,0,91,91,91,91
        ,91,91,91,91,91
DM 1010 DATA 0,7,0,75,73,74,85
        ,91,73,73,74,85
HJ 1020 DATA 0,11,0,96,81,96,8
        5,91,75,74,113,73
DM 1030 DATA 0,15,0,95,224,224
        ,96,95,224,96,96,95
GA 1040 DATA 0,13,0,85,96,73,9
        6,91,96,74,96,75
KG 1050 DATA 0,9,0,77,96,78,96
        ,86,96,78,96,77
ED 1060 DATA 0,9,0,96,96,96,79
        ,80,96,76,122,96
DR 1070 DATA 0,9,0,96,96,96,85
        ,73,96,74,75,96
QG 1080 REM INIT MAT TABLE
PE 1090 FOR X=0TO53:MAT(X,4)=0
        :NEXT
HC 1100 FOR X=0TO26
AF 1110 FOR Y=0TO1
DG 1120 LET R9=INT(RND(0)*54)
HF 1130 IF R9<0 OR R9>53 THEN
        {SPACE}GOTO1120
CD 1140 IF MAT(R9,4)=9 THEN GO
        TO1120
CM 1150 LET MAT(R9,0)=X:MAT(R9
        ,1)=0:LET MAT(R9,4)=9
GR 1160 MAT(R9,2)=00:MAT(R9,3)
        =SYM(X,1)
JR 1170 FOR E=3 TO 11
DG 1180 MAT(R9,E+2)=SYM(X,E)
QC 1190 NEXT E
DX 1200 NEXT Y
HX 1210 NEXT X
BJ 1220 K=1:K2=1:A=0:B=8
QS 1230 FOR W=0TO5
GE 1240 FOR X=ATOB
QD 1250 MAT(X,14)=K:MAT(X,15)=
        K2:K2=K2+4:NEXT X
BJ 1260 K=K+4:K2=1
GS 1270 A=A+9:B=B+9:NEXT W
CA 1280 RETURN
DS 1290 REM INPUT OPTIONS
XA 1300 LET JS=1
XD 1310 PRINT"{CLR}"
ED 1320 INPUT "{BLK}ONE OR TWO
         PLAYER GAME";PL
BQ 1330 IF PL<>1 AND PL<>2 THE
        N1310
HB 1340 IF PL=2 THEN INPUT"
        {CLR}{DOWN}ENTER NO. O
        F JOYSTICKS(1 OR 2)";J
        S
QE 1350 IF JS<>1 AND JS<>2 THE
        N1340
FC 1360 INPUT"ENTER NO. OF MAT
        CHES (1-27)";NM:NM=INT
        (NM):IFNM<1ORNM>27THEN
        1370
BJ 1370 IFNM<1ORNM>27THENPRINT
        "{HOME}{DOWN}":GOTO136
        0
EA 1380 PRINT"{CLR}{10 DOWN}
        {6 RIGHT}PLEASE STANDB
        Y FOR 45 SECONDS"
GR 1390 LET UP=0
SG 1400 RETURN
BD 1410 REM BLANK OUT BOX
XB 1420 POKE 6,6
CX 1430 X=MAT(BX,14):POKE 252,
        X
GJ 1440 X=MAT(BX,15):POKE 253,
        X
CA 1450 Y=96
ME 1460 FOR X=49414 TO 49422:P
        OKE X,Y:NEXT
JF 1470 SYS49674
GX 1480 RETURN
QR 1490 X=MAT(BX,3):POKE 6,X
DQ 1500 X=MAT(BX,3):POKE 6,X
SM 1510 X=MAT(BX,14):POKE 252,
        X
DH 1520 X=MAT(BX,15):POKE 253,
        X
AJ 1530 Y=5
RX 1540 FOR X=49414 TO 49422
XK 1550 Q=MAT(BX,Y):POKE X,Q:Y
        =Y+1:NEXT
BJ 1560 SYS49674
MC 1570 RETURN
HK 1580 SYS49426:RO=PEEK(252):
        CO=PEEK(253)
DX 1590 IF BT(RO,CO)=88 THEN G
        OTO1580
XD 1600 BX=BT(RO,CO)
FJ 1610 IF MAT(BX,1)=99 THEN G
        OTO1580
HH 1620 RETURN
ES 1630 DATA 0,0,0,0,0,0,0,102,1
        02,102,102,102,102,102
        ,102,102,0,0,0,76,122
JA 1640 DATA 193,24,166,251,18
        9,0,220,74,176,5,198,2
        52,76,70,193,24,74
XK 1650 DATA 176,5,230,252,76,
        70,193,24,74,176,5,198
        ,253,76,70,193,24,74
AA 1660 DATA 176,5,230,253,76,
        70,193,24,74,176,211,3
        2,251,193,96,169,36
XH 1670 DATA 197,253,208,4,162
        ,0,134,253,169,25,197,
        252,208,4,162,0,134
HQ 1680 DATA 252,24,169,0,101,
        253,16,4,162,35,134,25
        3,24,169,0,101,252
QD 1690 DATA 16,4,162,24,134,2
        52,173,5,193,201,0,240
        ,3,32,251,193,169,1
DB 1700 DATA 141,5,193,24,166,
        252,164,253,142,1,193,
        140,2,193,32,240,255
SM 1710 DATA 32,165,193,32,207
        ,193,169,113,32,210,25
        5,162,48,160,255,136
SB 1720 DATA 208,253,202,208,2
        48,76,21,193,165,252,1
        62,1,134,250,10,10
PQ 1730 DATA 101,252,10,10,38,
        250,10,38,250,101,253,
        133,249,169,0,101,250
JA 1740 DATA 133,250,174,0,193
        ,224,99,240,7,160,0,17
        7,249,141,4,193,96
HM 1750 DATA 162,54,134,248,16
        5,252,133,247,10,10,10
        1,247,10,10,38,248
JS 1760 DATA 10,38,248,101,253
        ,133,247,169,0,101,248
        ,133,248,174,0,193
QC 1770 DATA 224,99,240,7,160,
        0,177,247,141,3,193,96
        ,173,4,193,160,0,145
KS 1780 DATA 249,160,0,173,3,1
        93,145,247,96,162,99,1
        42,0,193,162,0,142
SE 1790 DATA 5,193,32,165,193,
        166,250,134,3,166,249,
        134,2,32,207,193,166
SF 1800 DATA 248,134,5,166,247
        ,134,4,162,0,142,0,193
        ,162,3,142,15,193,162
EA 1810 DATA 0,160,0,189,6,193
        ,145,2,165,6,145,4,200
        ,232,192,3,208,241
MH 1820 DATA 24,169,40,101,2,1
        33,2,169,0,101,3,133,3
        ,24,169,40,101,4,133
RS 1830 DATA 4,169,0,101,5,133
        ,5,206,15,193,160,0,20
        4,15,193,208,203,96
SR 1840 FOR X=49408TO49771:REA
        D A:POKE X,A:NEXT
DE 1850 RETURN
```

Program 2: Amiga Rememory

```
DEFINT a-z:DEFSNG r,g,b,mx:RANDO
MIZE TIMER:SCREEN 1,320,200,5,1
WINDOW 3,"",(0,0)-(311,186),16,1
:WINDOW OUTPUT 3
DIM bn(5,8),cb(26),r(11),b(11),d
f(7),aLt(7),hor(7),ver(7),sL(7),
rn(7),ck(7)
RESTORE PaletteData:FOR i=0 TO 1
5:READ r,g,b:PALETTE i,r,g,b:NEX
T
FOR i=20 TO 21:READ r,g,b:PALETT
E i,r,g,b:NEXT
FOR i=0 TO 5:READ r(i),b(i):PALE
TTE i+22,r(i),0,b(i):NEXT
FOR i=0 TO 5:r(11-i)=b(i):b(11-i
)=r(i):NEXT
PaletteData:
DATA 0,0,0,.5,.5,.5,.5,.5,.5,.6,
0,0
DATA 0,.6,0,.6,.6,0,.6,0,.6,0,.6
,.6
DATA 0,0,.6,.9,.9,.9,0,0,0,.9
,0
DATA 0,0,.9,.9,.9,0,.9,0,.9,.5,.
5,.5
DATA 0,0,0,0,0,0
DATA .6,0,.8,0,1,0,.8,0,.6,0,.5,
.3
FOR i=0 TO 26:READ cb(i):NEXT
DATA 9,4,13,15,3,0,9,6,12,9,4,20
,13,0,0,0,22,12,9,13,15,15,9,8,9
,3,10
e$=SPACE$(3):ON TIMER(1) GOSUB C
ycLe
FOR i=0 TO 7
df(i)=&HFFFF:ver(i)=&HAAAA:rn(i)
=RND*&HFFFF
IF (i AND 1) THEN hor(i)=&HFFFF:
aLt(i)=&HAAAA ELSE aLt(i)=&H5555
IF (i AND 4) THEN ck(i)=&HF0F0 E
LSE ck(i)=&H0F0F
NEXT
FOR i=0 TO 3:READ sL(i):sL(i+4)=
sL(i):NEXT
DATA &H3333,&H6666,&hcccc,&H9999
Start:
COLOR 2,0:CLS:GOSUB InPLayers:LO
CATE 13,9:GOSUB InMatches
GOSUB RandBoard:mf=0:ts=0:FOR i=
0 TO 1:tr(i)=0:sc(i)=0:NEXT
GOSUB DrawBoard:pL=0:m=1:sw=0:TI
MER ON
WHILE ts<nm
IF np THEN PALETTE 1,.6*(1-pL),.
6*pL,0
GOSUB SelectBox:GOSUB ShowPic:rl
=ro:cl=co:GOSUB SelectBox:GOSUB
ShowPic
IF bn(rl,cl)=bn(ro,co) THEN
sc(pL)=sc(pL)+1:ts=ts+1:m=0
ELSE
FOR i=1 TO 4000:NEXT:GOSUB HideP
ic
END IF
tr(pL)=tr(pL)+1:GOSUB UpdateScor
e:IF m THEN pL=pL XOR np
```

```
m=1
WEND
COLOR 15,0:LOCATE 9,10:PRINT "An
other game (y/n)?"
EndLp: k$=UCASE$(INKEY$):IF k$="
Y" THEN Start
IF k$="N" THEN TIMER OFF:SCREEN
CLOSE 1:WINDOW CLOSE 3:END ELSE
EndLp
'
SelectBox:
WHILE MOUSE(0)=0:WEND:px=MOUSE(1
):py=MOUSE(2):WHILE MOUSE(0)<>0:
WEND
IF POINT(px,py)<>2 THEN SelectBo
x
px=px AND &HFFE0:py=py AND &HFFE
0:ro=INT(py/32):co=INT(px/32):py
=py+8
RETURN
'
ShowPic:
n=bn(ro,co):COLOR ,cb(n):GOSUB H
ide
ON n+1 GOTO 1,1,1,2,3,4,5,6,7,8,
9,10,11,12,13,14,15,16,17,18,19,
20,21,22,23,24,1
LOCATE cy,cx:PRINT STR$(bn(ro,co
))
1 RETURN
2 COLOR 0:LOCATE cy+1,cx+1:PRINT
CHR$(214):PAINT(px+12,py+12):RET
URN
3 COLOR 13:a$=CHR$(191)+CHR$(63)
:LOCATE cy,cx:PRINT a$CHR$(191)
LOCATE cy+1,cx:PRINT CHR$(63)a$:
LOCATE cy+2,cx:PRINT a$CHR$(191)
;:RETURN
4 PATTERN ,sL:FOR i=0 TO 7:COLOR
i+7
LINE(px+2*i,py+2*i)-(px+23-2*i,p
y+23-2*i),,bf:NEXT:PATTERN ,df:R
ETURN
5 COLOR 10:AREA(px+2,py+12):AREA
STEP(10,-10):AREA STEP(10,10)
AREA STEP(-10,10):AREA STEP(-10,
-10):PATTERN ,hor:AREAFILL:PATTE
RN ,df:RETURN
6 COLOR 14:GOSUB Triangle:AREAFI
LL:RETURN
7 COLOR 13:CIRCLE(px+19,py+5),2:
PAINT(px+20,py+5)
PATTERN ,sL:GOSUB Triangle:AREAF
ILL:PATTERN ,df:RETURN
8 COLOR ,3:LOCATE cy,cx+1:PRINT
SPACE$(1):LOCATE cy+1,cx:PRINT e
$
LOCATE cy+2,cx+1:PRINT SPACE$(1)
;:RETURN
9 FOR i=0 TO 2:FOR j=0 TO 2:COLO
R 20+((i+j) AND 1):x=px+5*i+7:y=
py+5*j+7
CIRCLE(x,y),2:NEXT j,i:RETURN
10 COLOR 21:GOSUB Box:PATTERN ,a
Lt:AREAFILL:PATTERN ,df:RETURN
11 COLOR 6:GOSUB Box:PATTERN ,sL
:AREAFILL:PATTERN ,df:RETURN
12 COLOR 14:PATTERN ,hor:x=px+4:
y=py+4:GOSUB Diamond:AREAFILL
x=px:y=py+12:GOSUB Diamond:AREAF
ILL
x=px+12:y=py+16:GOSUB Diamond:AR
EAFILL:PATTERN ,df:RETURN
13 FOR i=0 TO 11:COLOR (i MOD 6)
+22:LINE(px,py+2*i)-(px+23,py+23
-2*i):NEXT
RETURN
14 FOR i=0 TO 11:COLOR (i MOD 6)
+22:LINE(px,py)-(px+2*i,py+23-2*
i)
LINE(px+23,py+23)-(px+2*i,py+23-
2*i):NEXT:RETURN
15 COLOR 24:GOSUB Box:PATTERN ,a
Lt:AREAFILL:COLOR ,0
LOCATE cy+1,cx+1:PRINT SPACE$(1)
:PATTERN ,df:RETURN
16 COLOR 20:x=px+8:y=py+12:GOSUB
Diamond:AREAFILL:x=x+4
CIRCLE(x,y),2,21:PAINT(x,y),21,2
1:RETURN
17 COLOR 12:GOSUB Triangle:PATTE
RN ,ver:AREAFILL:PATTERN ,hor:CO
LOR 9,12
AREA(px,py):AREA STEP(23,0):AREA
STEP(0,23):AREA STEP(-23,-23)
AREAFILL:PATTERN ,df:RETURN
18 PATTERN ,rn:PAINT(px,py),22,0
:PATTERN ,df:RETURN
19 PATTERN ,ck:PAINT(px,py),0,0:
PATTERN ,df:RETURN
20 COLOR 15,0:LINE(px,py+7)-(px+
23,py+7),0
LOCATE cy+1,cx:PRINT CHR$(240)CH
R$(245)CHR$(240):RETURN
21 FOR i=1 TO 11:LINE(px,py+i*2)
-(px+23,py),10
LINE(px,py+23)-(px+23,py+i*2),12
:NEXT:RETURN
22 FOR i=4 TO 20:LINE(px,py+i)-(
px+23,py+i),(i MOD 4)+28:NEXT:RE
TURN
23 COLOR 0:LINE(px,py+17)-(px+6,
py+9):LINE -STEP(6,2):LINE -STEP
(6,6)
LINE -STEP(5,0):PAINT(px,py),0,0
:LINE(px,py+16)-(px+6,py+8),28
LINE -STEP(6,2),29:LINE -STEP(6,
6),30:LINE -STEP(5,0),31:RETURN
24 LINE(px,py+8)-(px+23,py+16),0
:PAINT(px,py+23),5,0:RETURN
'
Triangle:
AREA(px,py):AREA STEP(23,23):ARE
A STEP(-23,0):AREA STEP(0,-23):R
ETURN
'
Diamond:
AREA(x,y):AREA STEP(4,-4):AREA S
TEP(4,4):AREA STEP(-4,4):AREA ST
EP(-4,-4)
RETURN
'
Box:
AREA(px,py):AREA STEP(23,0):AREA
STEP(0,23):AREA STEP(-23,0)
AREA STEP(0,-23):RETURN
'
HidePic:
COLOR ,2:GOSUB Hide:ro=r1:co=c1:
GOSUB Hide:RETURN
Hide: cx=4*co+1:cy=4*ro+2
FOR i=0 TO 2:LOCATE cy+i,cx:PRIN
T e$;:NEXT:RETURN
'
UpdateScore:
COLOR 0,pL+3:pr=8*pL-4*np+13:LOC
ATE pr,37:s$=STR$(tr(pL))
PRINT RIGHT$("00"+RIGHT$(s$,LEN(
s$)-1),3)
LOCATE pr+2,37:s$=STR$(sc(pL))
PRINT RIGHT$("00"+RIGHT$(s$,LEN(
s$)-1),3)
RETURN
'
DrawBoard:
CLS:COLOR ,2:FOR i=0 TO 23
IF (i AND 3)<>0 THEN
FOR j=0 TO 8:PRINT e$SPC(1);:NEX
T
END IF
IF i<23 THEN PRINT
NEXT
FOR pL=0 TO np:COLOR 0,pL+3:FOR
j=0 TO 6:LOCATE 8*pL-4*np+10+j,3
7
PRINT e$:NEXT:LOCATE 8*pL-4*np+1
1,37:PRINT STR$(pL+1):GOSUB Upda
teScore:NEXT
RETURN
'
RandBoard:
i=0:FOR j=0 TO 4 STEP 2:FOR k=0
TO 8:bn(j,k)=i:bn(j+1,k)=i:i=i+1
:NEXT k,j
FOR j=0 TO 5:FOR k=0 TO 8:sj=INT
(RND*5):sk=INT(RND*9)
t=bn(sj,sk):bn(sj,sk)=bn(j,k):bn
(j,k)=t:NEXT k,j
RETURN
'
InPLayers:
LOCATE 11,9:PRINT "Number of pla
yers (1/2)?"
GetKey:k$=INKEY$:IF k$="" OR (k$
<>"1" AND k$<>"2") THEN GetKey
np=VAL(k$)-1
RETURN
'
InMatches:
INPUT "Number of matches (1-27)?
",s$
nm=VAL(s$):IF nm<1 OR nm>27 THEN
nm=27
RETURN
'
CycLe:
nsw=sw XOR 1:PALETTE 20,0,sw*.9,
0:PALETTE 21,0,nsw*.9,0
sw=(sw+1) MOD 2:cc=(cc+1) MOD 12
'
FOR cn=28 TO 31:PALETTE cn,1,1,1
:NEXT:PALETTE (cc MOD 4)+28,0,0,
1
FOR cn=0 TO 5:ck=(cc+cn) MOD 12:
PALETTE cn+22,r(ck),0,b(ck):NEXT
'
RETURN
```

Program 3: IBM PC/PCjr Rememory

```
BM 10 KEY OFF:DEF SEG=0:DEFINT A
      -Z:POKE 1047,PEEK(1047) OR
       64:RANDOMIZE TIMER
GA 20 SCREEN 0,1:WIDTH 40:LOCATE
      ,,0:COLOR 8,0,0:CLS
FM 30 DIM CF(26),CB(26),PS$(26,2
      ),BN(5,8):GOSUB 1500:GOSUB
       4000:LOCATE 13,9:GOSUB 45
      00
DA 40 GOSUB 3000:RO=0:CO=0:PX=1:
      PY=1:MF=0:TS=0:FOR I=0 TO
      1:TR(I)=0:SC(I)=0:NEXT:GOS
      UB 1000:PL=0
LJ 45 WHILE TS<NM
KN 50 GOSUB 2000:IF BN(RO,CO)=27
       THEN 50 ELSE GOSUB 1200:R
      1=RO:C1=CO
OO 60 GOSUB 2000:IF (BN(RO,CO)=2
      7) OR ((R1=RO) AND (C1=CO)
      ) THEN 60 ELSE GOSUB 1200
JK 70 IF BN(R1,C1)=BN(RO,CO) THE
      N SC(PL)=SC(PL)+1:TS=TS+1:
      BN(RO,CO)=27:BN(R1,C1)=27
      ELSE FOR I=1 TO 2000:NEXT:
      GOSUB 1100
DL 80 TR(PL)=TR(PL)+1:GOSUB 1070
      :IF BN(RO,CO)<27 THEN PL=P
      L XOR NP
GO 90 WEND
BA 100 COLOR 7,0:LOCATE 9,10:PRI
       NT "Another game (Y/N)?"
HB 110 K$=INKEY$:IF K$="Y" THEN
       CLS:LOCATE 13,7:GOSUB 450
       0:COLOR 8:GOTO 40
NH 120 IF K$="N" THEN CLS:END EL
       SE 110
NE 1000 E$=STRING$(3,219)
MG 1010 FOR I=0 TO 23:LOCATE ,2
MA 1020 IF (I AND 3)<>0 THEN FOR
        J=0 TO 8:PRINT E$SPC(1)
       ;:NEXT
GA 1030 IF I<23 THEN PRINT
OI 1040 NEXT
EB 1050 FOR PL=0 TO NP:COLOR 0,P
       L*2+2:FOR J=0 TO 6:LOCAT
       E 8*PL-4*NP+10+J,38:PR
```

```
      T SPACE$(3):NEXT:LOCATE
      8*PL-4*NP+11,38:PRINT ST
      R$(PL+1):GOSUB 1070:NEXT
JN 1060 RETURN
IB 1070 COLOR 0,PL*2+2:PR=8*PL-4
      *NP+13:LOCATE PR,38:S$=S
      TR$(TR(PL)):PRINT RIGHT$
      ("00"+RIGHT$(S$,LEN(S$)-
      1),3)
PC 1080 LOCATE PR+2,38:S$=STR$(S
      C(PL)):PRINT RIGHT$("00"
      +RIGHT$(S$,LEN(S$)-1),3)
JG 1090 RETURN
OJ 1100 COLOR 8:GOSUB 1150:R1=RO
      :C1=CO:GOSUB 1150:RETURN
LM 1150 X=4*C1+2:Y=4*R1+2:LOCATE
       Y,X:PRINT E$DL$E$DL$E$;
      :RETURN
JE 1200 LOCATE PY+1,PX+1:N=BN(RO
      ,CO):COLOR CF(N),CB(N):P
      RINT PS$(N,0)DL$PS$(N,1)
      DL$PS$(N,2);:RETURN
FB 1500 DL$=CHR$(31)+STRING$(3,2
      9)
ML 1510 FOR I=0 TO 26
0M 1520 READ CF(I),CB(I):FOR J=0
       TO 2:READ T0,T1,T2:PS$(
      I,J)=CHR$(T0)+CHR$(T1)+C
      HR$(T2):NEXT J,I
JO 1530 RETURN
CP 1800 DATA 6,1,168,63,168,63,1
      68,63,168,63,168
PO 1805 DATA 7,5,201,202,187,211
      ,210,210,218,208,215
NM 1810 DATA 14,4,32,32,32,32,15
      ,32,32,32,32
0M 1815 DATA 9,2,15,15,15,15,178
      ,15,15,15,15
KN 1820 DATA 4,7,244,244,159,245
      ,179,244,159,245,245
DP 1825 DATA 8,2,177,176,177,176
      ,177,176,177,176,177
PF 1830 DATA 8,0,223,223,223,6,6
      ,6,220,220,220
QN 1835 DATA 13,1,32,32,32,157,3
      2,157,32,157,32
GF 1840 DATA 0,7,176,176,176,176
      ,176,176,176,176,176
GM 1845 DATA 10,2,32,4,32,4,32,4
      ,32,4,32
NI 1850 DATA 4,7,32,219,32,219,2
      19,219,32,219,32
OH 1855 DATA 7,3,178,178,178,178
      ,178,178,178,178,178
JF 1860 DATA 0,6,206,206,206,206
      ,206,206,206,206,206
0M 1865 DATA 0,4,32,32,32,32,32,
      32,32,32,32
ME 1870 DATA 14,0,219,219,219,21
      9,219,219,219,219,219
DF 1875 DATA 3,1,247,247,247,247
      ,247,247,247,247,247
MM 1880 DATA 12,4,222,186,221,24
      0,240,240,222,186,221
QP 1885 DATA 4,0,32,95,32,248,32
      ,248,92,236,47
ML 1890 DATA 8,5,248,248,248,248
      ,248,248,248,248,248
NL 1895 DATA 0,2,32,32,32,32,32,
      32,32,32,32
FB 1900 DATA 12,5,177,177,177,17
      7,177,177,177,177,177
MD 1905 DATA 4,7,248,249,248,250
      ,249,248,250,249,250
LF 1910 DATA 15,7,32,32,237,32,2
      37,32,237,32,32
BK 1915 DATA 12,1,184,64,213,192
      ,197,217,214,193,183
FG 1920 DATA 13,4,232,32,232,32,
      32,32,232,32,232
DI 1925 DATA 1,7,14,32,32,32,32,
      32,32,251,32
BC 1930 DATA 10,1,188,32,200,32,
      234,32,187,32,201
FH 2000 GOSUB 2500
ID 2005 K$=RIGHT$(INKEY$,1):IF K
      $="" THEN 2005 ELSE K=AS
      C(K$)
FH 2010 IF K=13 THEN LOCATE PY,P
      X:PRINT SPACE$(5);:LOCAT
      E PY+4,PX:PRINT SPACE$(5
      );:RETURN
AE 2020 IF K=72 THEN IF RO>0 THE
      N RO=RO-1:GOSUB 2500
QN 2030 IF K=80 THEN IF RO<5 THE
      N RO=RO+1:GOSUB 2500
PH 2040 IF K=75 THEN IF CO>0 THE
      N CO=CO-1:GOSUB 2500
GA 2050 IF K=77 THEN IF CO<8 THE
      N CO=CO+1:GOSUB 2500
NK 2060 GOTO 2005
FL 2500 X=4*CO+1:Y=4*RO+1:COLOR
      PL*2+2,0
EO 2510 LOCATE PY,PX:PRINT SPACE
      $(5);:LOCATE PY+4,PX:PRI
      NT SPACE$(5);
KL 2520 LOCATE Y,X:PRINT CHR$(21
      8)SPC(3)CHR$(191);:LOCAT
      E Y+4,X:PRINT CHR$(192)S
      PC(3)CHR$(217);
LE 2530 PX=X:PY=Y
JC 2540 RETURN
IE 3000 I=0:FOR J=0 TO 4 STEP 2:
      FOR K=0 TO 8:BN(J,K)=I:B
      N(J+1,K)=I:I=I+1:NEXT K,
      J
DG 3010 FOR J=0 TO 5:FOR K=0 TO
      8:SJ=INT(RND*5):SK=INT(R
      ND*9)
MA 3020 T=BN(SJ,SK):BN(SJ,SK)=BN
      (J,K):BN(J,K)=T:NEXT K,J
IG 3030 RETURN
BF 4000 LOCATE 11,9:PRINT "Numbe
      r of players (1/2)?"
KD 4010 K$=INKEY$:IF K$="" OR (K
      $<>"1" AND K$<>"2") THEN
       4010
HJ 4020 NP=VAL(K$)-1
IH 4030 RETURN
MD 4500 INPUT "Number of matches
       (1-27)? ",S$
EH 4510 NM=VAL(S$):IF NM<1 OR NM
      >27 THEN NM=27
JO 4520 RETURN
```

Program 4: Apple II Rememory

```
09 20 HOME : GOSUB 4000: VTAB 13
      : HTAB 5: GOSUB 4500
52 30 DIM PS$(26,2),BN(5,8): GOS
      UB 1500: GOSUB 10000
DC 40 GOSUB 3000:RO = 0:CO = 0:P
      X = 1:PY = 1:MF = 0:TS = 0
      : FOR I = 0 TO 1:TR(I) = 0
      :SC(I) = 0: NEXT : GOSUB 1
      000:PL = 0: GOSUB 1095
19 50 GOSUB 2000: IF BN(RO,CO) =
       27 THEN 50
53 55 GOSUB 1200:R1 = RO:C1 = CO
B0 60 GOSUB 2000: IF (BN(RO,CO)
      = 27) OR ((R1 = RO) AND (C
      1 = CO)) THEN 60
18 65 GOSUB 1200
E3 70 IF BN(R1,C1) = BN(RO,CO) T
      HEN SC(PL) = SC(PL) + 1:TS
       = TS + 1:BN(RO,CO) = 27:B
      N(R1,C1) = 27: GOTO 80
D7 75 FOR I = 1 TO 1000: NEXT :
      GOSUB 1100
4D 80 TR(PL) = TR(PL) + 1: GOSUB
       1070: IF BN(RO,CO) < 27 T
      HEN PL = NP - PL
61 90 IF TS < NM THEN GOSUB 1095
      : GOTO 50
DD 100 VTAB 8: HTAB 8: PRINT "AN
      OTHER GAME (Y OR N)? ";
EA 110 GET K$: IF K$ = "Y" THEN
      VTAB 12: HTAB 4: GOSUB 45
      00: HGR2 : GOTO 40
A4 120 IF K$ = "N" THEN HOME : E
      ND
91 130 GOTO 110
95 1000 HOME :E$ = "": FOR I = 0
       TO 2:E$ = E$ + CHR$ (32
      ): NEXT
A6 1010 K = 0: FOR I = 0 TO 23:
      HTAB 2: IF K = 4 THEN K
      = 0
B8 1020 IF K < > 3 THEN FOR J =
      0 TO 8: INVERSE : PRINT
      E$;: NORMAL : PRINT SPC(
       1);: NEXT
EF 1030 IF I < 23 THEN PRINT
61 1040 K = K + 1: NEXT
B6 1050 FOR PL = 0 TO NP: INVERS
      E : FOR J = 0 TO 6: VTAB
       8 * PL - 4 * NP + 9 + J
      : HTAB 38: PRINT E$: NEX
      T : VTAB 8 * PL - 4 * NP
       + 10: HTAB 39: PRINT ST
      R$ (PL + 1): GOSUB 1070:
       NEXT
E9 1060 RETURN
A6 1070 INVERSE :PR = 8 * PL - 4
       * NP + 12: VTAB PR: HTA
      B 38:S$ = STR$ (TR(PL)):
       PRINT RIGHT$ ("00" + S$
      ,3)
8D 1080 VTAB PR + 2: HTAB 38:S$
      = STR$ (SC(PL)): PRINT R
      IGHT$ ("00" + S$,3)
83 1090 NORMAL : RETURN
DB 1095 INVERSE : VTAB 8 * NP *
      (1 - PL) - 4 * NP + 10:
      HTAB 38: PRINT CHR$ (32)
      :PR = 8 * PL - 4 * NP +
      10: VTAB PR: HTAB 38: PR
      INT CHR$ (105): NORMAL :
       RETURN
0F 1100 INVERSE : GOSUB 1150:R1
      = RO:C1 = CO: GOSUB 1150
      : NORMAL : RETURN
3C 1150 X = 4 * C1 + 2:Y = 4 * R
      1 + 1: VTAB Y: HTAB X: P
      RINT E$: HTAB X: PRINT E
      $: HTAB X: PRINT E$;: RE
      TURN
BB 1200 FOR J = 0 TO 2: VTAB PY
      + J: HTAB PX + 1: PRINT
      PS$(BN(RO,CO),J);: NEXT
      : RETURN
EB 1500 FOR I = 0 TO 26: FOR J =
       0 TO 2
38 1510 PS$(I,J) = STR$ (I): NEX
      T J,I
F3 1520 FOR I = 0 TO 26: FOR J =
       0 TO 2
F5 1530 READ T0,T1,T2:PS$(I,J) =
      CHR$ (T0) + CHR$ (T1) +
      CHR$ (T2): NEXT J,I
EB 1540 RETURN
25 1800 DATA 35,35,35,35,35,35,3
      5,35,35
31 1810 DATA 58,63,58,63,58,63,5
      8,63,58
71 1820 DATA 64,64,43,64,43,32,4
      3,32,32
D5 1830 DATA 32,44,64,42,32,44,6
      4,42,32
A4 1840 DATA 33,32,34,32,35,32,3
      4,32,33
84 1850 DATA 47,46,46,47,46,46,4
      6,47,47
4E 1860 DATA 36,38,37,32,39,36,3
      9,36,39
17 1870 DATA 64,91,64,92,91,92,6
      4,91,64
40 1880 DATA 61,61,61,61,61,61,6
      1,61,61
47 1890 DATA 91,91,91,91,91,91,9
```

```
      1,91,91
9F 1900 DATA 92,91,92,64,64,64,9
      1,92,91
48 1905 DATA 45,32,42,32,94,32,4
      4,32,43
6E 1910 DATA 39,32,37,32,93,32,3
      8,32,36
82 1915 DATA 36,37,38,39,59,37,3
      7,38,39
32 1920 DATA 46,46,46,46,46,46,4
      6,46,46
EA 1925 DATA 94,93,94,94,93,94,9
      4,93,94
A1 1930 DATA 91,92,91,91,92,91,9
      1,92,91
EA 1935 DATA 92,91,92,92,91,92,9
      2,91,92
3D 1940 DATA 95,95,95,95,95,95,9
      5,95,95
5D 1945 DATA 32,32,32,96,96,96,3
      2,32,32
45 1950 DATA 97,97,97,97,97,97,9
      7,97,97
EE 1955 DATA 98,33,33,34,34,34,9
      9,33,99
E4 1960 DATA 34,33,100,100,34,33
      ,34,33,100
2F 1965 DATA 32,96,32,39,59,36,3
      8,59,37
F4 1970 DATA 32,39,37,32,38,36,3
      2,32,32
77 1975 DATA 101,102,101,101,102
      ,101,101,102,101
7D 1980 DATA 103,104,103,103,104
      ,103,103,104,103
42 2000 GOSUB 2500
7E 2010 GET K$: IF K$ = CHR$ (32
      ) THEN VTAB PY: HTAB PX:
       PRINT CHR$ (32);: HTAB
      PX + 4: PRINT CHR$ (32):
       VTAB PY + 2: HTAB PX: P
      RINT CHR$ (32);: HTAB PX
       + 4: PRINT CHR$ (32): R
      ETURN
29 2020 IF K$ = "I" THEN IF RO >
       0 THEN RO = RO - 1: GOS
      UB 2500
BC 2030 IF K$ = "M" THEN IF RO <
      5 THEN RO = RO + 1: GOS
      UB 2500
DA 2040 IF K$ = "J" THEN IF CO >
       0 THEN CO = CO - 1: GOS
      UB 2500
C2 2050 IF K$ = "K" THEN IF CO <
       8 THEN CO = CO + 1: GOS
      UB 2500
68 2060 GOTO 2010
86 2500 X = 4 * CO + 1:Y = 4 * R
      O + 1
9F 2510 VTAB PY: HTAB PX: PRINT
      CHR$ (32);: HTAB PX + 4:
       PRINT CHR$ (32): VTAB P
      Y + 2: HTAB PX: PRINT CH
      R$ (32);: HTAB PX + 4: P
      RINT CHR$ (32);
2A 2520 VTAB Y: HTAB X: PRINT CH
      R$ (62);: HTAB X + 4: PR
      INT CHR$ (60): VTAB Y +
      2: HTAB X: PRINT CHR$ (6
      2);: HTAB X + 4: PRINT C
      HR$ (60);
C5 2530 PX = X:PY = Y
EC 2540 RETURN
6C 3000 I = 0: FOR J = 0 TO 4 ST
      EP 2: FOR K = 0 TO 8:BN(
      J,K) = I:BN(J + 1,K) = I
      :I = I + 1: NEXT K,J
27 3010 FOR J = 0 TO 5: FOR K =
      0 TO 8:SJ = INT ( RND (1
      ) * 5):SK = INT ( RND (1
      ) * 9)
B6 3020 T = BN(SJ,SK):BN(SJ,SK)
      = BN(J,K):BN(J,K) = T: N
      EXT K,J
DF 3030 RETURN
66 4000 VTAB 11: HTAB 5: PRINT "
      NUMBER OF PLAYERS (1 OR
      2)? ";
0F 4005 K = PEEK (49152):R = RND
      (1): IF K > 127 THEN PO
      KE 49168,0
CA 4010 IF K < > 177 AND K < > 1
      78 THEN 4005
70 4020 NP = K - 177
E0 4030 RETURN
15 4500 INPUT "NUMBER OF MATCHES
      (1 TO 27)? ";S$
D5 4510 NM = VAL (S$): IF NM < 1
      OR NM > 27 THEN NM = 27
E6 4520 RETURN
3A 10000 HGR2 : GOSUB 10990: GOS
      UB 12000
D0 10030 IF PEEK (190 * 256) = 7
      6 THEN PRINT CHR$ (4);"
      PR#A768": GOTO 10050
F9 10040 POKE 54,0: POKE 55,3: C
      ALL 1002
FC 10050 POKE 6,0: POKE 7,128: P
      OKE 230,64
83 10060 RETURN
F2 10990 FOR I = 768 TO 855: REA
      D A: POKE I,A: NEXT : R
      ETURN
F3 11000 DATA 216,120,133,69,134
      ,70,132,71
5E 11010 DATA 166,7,10,10,176,4,
      16,62
6B 11020 DATA 48,4,16,1,232,232,
      10,134
A3 11030 DATA 27,24,101,6,133,26
      ,144,2
97 11040 DATA 230,27,165,40,133,
      8,165,41
85 11050 DATA 41,3,5,230,133,9,1
      62,8
56 11060 DATA 160,0,177,26,36,50
      ,48,2
88 11070 DATA 73,127,164,36,145,
      8,230,26
E4 11080 DATA 208,2,230,27,165,9
      ,24,105
7A 11090 DATA 4,133,9,202,208,22
      6,165,69
3E 11100 DATA 166,70,164,71,88,7
      6,240,253
F2 12000 FOR I = 32768 TO 33359:
       READ A: POKE I,A: NEXT
       : RETURN
50 12010 DATA 0,0,0,0,0,0,0,0
46 12020 DATA 1,3,6,12,24,48,96,
      64
3B 12030 DATA 64,96,48,24,12,6,3
      ,1
DF 12040 DATA 65,99,54,28,28,54,
      99,65
25 12050 DATA 152,152,152,143,13
      5,128,128,128
C3 12060 DATA 128,128,128,135,14
      3,152,152,152
4C 12070 DATA 140,140,140,248,24
      0,128,128,128
45 12080 DATA 128,128,128,240,24
      8,140,140,140
08 12090 DATA 128,176,152,140,14
      0,152,176,128
A1 12100 DATA 128,134,140,152,15
      2,140,134,128
BE 12110 DATA 0,1,3,7,15,31,63,1
      27
55 12120 DATA 127,63,31,15,7,3,1
      ,0
02 12130 DATA 127,126,124,120,11
      2,96,64,0
CC 12140 DATA 0,64,96,112,120,12
      4,126,127
AF 12150 DATA 25,51,102,76,25,51
      ,102,76
1E 12160 DATA 76,102,51,25,76,10
      2,51,25
3B 12170 DATA 128,188,230,246,23
      8,230,188,128
0B 12180 DATA 128,152,156,152,15
      2,152,188,128
83 12190 DATA 128,188,230,176,14
      0,230,254,128
83 12200 DATA 128,188,230,176,22
      4,230,188,128
95 12210 DATA 128,176,184,180,25
      4,176,176,128
8C 12220 DATA 128,254,134,190,22
      4,230,188,128
1B 12230 DATA 128,188,134,190,23
      0,230,188,128
E6 12240 DATA 128,254,224,176,15
      2,140,140,128
D0 12250 DATA 128,188,230,188,23
      0,230,188,128
73 12260 DATA 128,188,230,230,25
      2,176,152,128
D1 12270 DATA 128,152,128,152,17
      6,230,188,128
08 12280 DATA 140,140,140,255,25
      5,140,140,140
37 12290 DATA 128,152,140,134,14
      0,152,128,128
1E 12300 DATA 157,178,149,181,12
      8,173,205,186
A2 12310 DATA 128,140,152,176,15
      2,140,128,128
F8 12320 DATA 128,188,230,176,15
      2,128,152,128
17 12330 DATA 255,255,255,255,25
      5,255,255,255
D9 12340 DATA 128,252,230,230,25
      4,230,230,128
D7 12350 DATA 128,190,230,230,19
      0,230,254,128
7B 12360 DATA 128,188,230,134,13
      4,230,190,128
BF 12370 DATA 128,190,230,230,23
      0,230,190,128
F2 12380 DATA 128,254,134,134,19
      0,134,254,128
F0 12390 DATA 128,254,134,134,19
      0,134,134,128
65 12400 DATA 128,188,230,134,24
      6,230,190,128
44 12410 DATA 128,230,230,230,25
      4,230,230,128
8A 12420 DATA 128,152,152,152,15
      2,152,152,128
3A 12430 DATA 128,224,224,224,22
      4,230,188,128
9C 12440 DATA 128,230,230,182,15
      8,230,230,128
46 12450 DATA 128,134,134,134,13
      4,134,254,128
6C 12460 DATA 128,230,254,230,23
      0,230,230,128
AD 12470 DATA 128,190,230,230,23
      0,230,230,128
CB 12480 DATA 128,188,230,230,23
      0,230,188,128
D3 12490 DATA 128,190,230,230,19
      0,134,134,128
8F 12500 DATA 128,188,230,230,23
      0,182,236,128
97 12510 DATA 128,190,230,230,19
      0,230,230,128
C1 12520 DATA 128,188,230,140,17
      6,230,190,128
B7 12530 DATA 128,254,152,152,15
      2,152,152,128
4E 12540 DATA 128,230,230,230,23
      0,230,190,128
56 12550 DATA 128,230,230,230,23
      0,230,152,128
CA 12560 DATA 128,230,230,230,23
      0,254,230,128
A2 12570 DATA 128,230,230,230,18
      8,230,230,128
22 12580 DATA 128,230,230,230,18
      8,152,152,128
8E 12590 DATA 128,254,176,152,14
      0,134,254,128
```

```
F9 12600 DATA 170,170,170,170,17
       0,170,170,170
8A 12610 DATA 213,213,213,213,21
       3,213,213,213
C5 12620 DATA 162,162,170,136,13
       6,170,162,162
23 12630 DATA 136,136,170,162,16
       2,170,136,136
B3 12640 DATA 85,127,42,0,85,127
       ,42,0
2E 12650 DATA 0,8,42,42,42,42,8,
       0
B8 12660 DATA 74,109,85,74,106,7
       3,123,43
43 12670 DATA 96,96,96,96,96,96,
       96,96
A0 12680 DATA 3,3,3,3,3,3,3,3
4B 12690 DATA 0,0,0,0,0,0,0,127
8F 12700 DATA 85,85,85,85,85,85,
       85,85
17 12710 DATA 42,42,42,42,42,42,
       42,42
C8 12720 DATA 0,8,34,8,34,8,34,8
D0 12730 DATA 0,8,34,8,34,8,34,8
D3 12740 DATA 8,42,28,8,28,42,8,
       0
```

Program 5: Rememory For Atari 400, 800, XL, And XE

```
BJ 1  OPEN #1,4,0,"K:":PRINT
      CHR$(125)
IL 2  DIM SPACE1$(40),SPACE2$
      (20),BN(5,8),S$(6),TR(1
      ),SC(1)
HE 5  GOSUB 4000:POSITION 9,1
      3:GOSUB 4500
HN 10 GRAPHICS 0:POKE 82,0:P
      OKE 83,39
FH 15 POKE 752,1:PRINT
DE 20 DLIST=PEEK(560)+PEEK(5
      61)*256
CD 30 FOR T=6 TO 28 STEP 2:P
      OKE DLIST+T,6:NEXT T
PL 40 POKE 756,226
CB 50 SCREEN=PEEK(88)+PEEK(8
      9)*256
NK 60 FOR I=0 TO 660 STEP 60
      :K=SCREEN+I
MA 80 FOR J=40 TO 59:POKE K+
      J,128:NEXT J:NEXT I
PA 90 POKE 710,0:POKE 752,1
GO 105 GOSUB 3000:ROW=0:COL=
       0:MF=0:TS=0:FOR I=0 T
       O 1:TR(I)=0:SC(I)=0:N
       EXT I:GOSUB 1000:PL=0
OH 110 GOSUB 2000:IF BN(ROW,
       COL)=27 THEN 110
BE 115 GOSUB 1200:PROW=ROW:P
       COL=COL
GH 120 GOSUB 2000:IF (BN(ROW
       ,COL)=27) OR ((PROW=R
       OW) AND (PCOL=COL)) T
       HEN 120
NL 125 GOSUB 1200
OD 130 IF BN(PROW,PCOL)=BN(R
       OW,COL) THEN SC(PL)=S
       C(PL)+1:TS=TS+1:BN(RO
       W,COL)=27:BN(PROW,PCO
       L)=27:GOTO 140
KK 135 FOR I=1 TO 200:NEXT I
       :GOSUB 1100
JH 140 TR(PL)=TR(PL)+1:GOSUB
        1080:IF BN(ROW,COL)<
       27 THEN PL=NP-PL
OG 150 GOSUB 1070:IF TS<NM T
       HEN 110
DN 160 POSITION 10,9:PRINT "
       ANOTHER GAME (Y/N)?":
       POKE SCREEN+40,128
JF 170 GET #1,K:IF K=89 THEN
        GRAPHICS 0:PRINT CHR
       $(125):RUN
JB 180 IF K=78 THEN GRAPHICS
       0:PRINT CHR$(125):EN
       D
GL 190 GOTO 170
IK 1000 POS=SCREEN+63:FOR I=
       0 TO 5:FOR J=0 TO 8:
       POKE 4*J+POS,128:POK
       E 4*J+POS+1,128:NEXT
        J:POS=POS+120:NEXT
       I
IL 1050 POSITION 1,0:PRINT "
       *P1 000 000":IF NP=1
        THEN POSITION 29,0:
       PRINT "P2 000 000"
ND 1060 GOTO 1099
GC 1070 POSITION 27*PL+1,0:P
       RINT "*":POSITION 27
       *(1-PL)+1,0:PRINT CH
       R$(32):GOTO 1099
OI 1080 PR=27*PL+8:S$=STR$(T
       R(PL)):POSITION PR-L
       EN(S$),0:PRINT S$
IN 1090 S$=STR$(SC(PL)):POSI
       TION PR+4-LEN(S$),0:
       PRINT S$
DC 1099 POKE SCREEN+40,128:R
       ETURN
JK 1100 PIC=27:GOSUB 1150:PR
       OW=ROW:PCOL=COL:GOSU
       B 1150:RETURN
EM 1150 XPOS=PCOL:YPOS=PROW:
       GOSUB 20000:RETURN
NO 1200 PIC=BN(ROW,COL):XPOS
       =COL:YPOS=ROW:GOSUB
       20000:RETURN
AK 2000 GOSUB 2510
AM 2005 JY=STICK(PL*JS):POKE
        77,0
JA 2010 IF STRIG(PL*JS)=0 TH
       EN POKE PCUR,PC1:POK
       E PCUR+3,PC2:RETURN
FH 2020 IF JY=14 THEN IF ROW
       >0 THEN ROW=ROW-1:GO
       SUB 2500
PI 2030 IF JY=13 THEN IF ROW
       <5 THEN ROW=ROW+1:GO
       SUB 2500
KI 2040 IF JY=11 THEN IF COL
       >0 THEN COL=COL-1:GO
       SUB 2500
IC 2050 IF JY=7 THEN IF COL<
       8 THEN COL=COL+1:GOS
       UB 2500
GJ 2060 FOR I=0 TO 50:NEXT I
       :GOTO 2005
BC 2500 POKE PCUR,PC1:POKE P
       CUR+3,PC2
BO 2510 CUR=SCREEN+ROW*120+4
       *COL+62:PC1=PEEK(CUR
       ):PC2=PEEK(CUR+3)
FA 2520 POKE CUR,32:POKE CUR
       +3,32:PCUR=CUR
KK 2530 RETURN
FG 3000 I=0:FOR J=0 TO 4 STE
       P 2:FOR K=0 TO 8:BN(
       J,K)=I:BN(J+1,K)=I:I
       =I+1:NEXT K:NEXT J
BL 3010 FOR J=0 TO 5:FOR K=0
        TO 8:SJ=INT(RND(0)*
       5):SK=INT(RND(0)*9)
PD 3020 T=BN(SJ,SK):BN(SJ,SK
       )=BN(J,K):BN(J,K)=T:
       NEXT K:NEXT J
KG 3030 RETURN
LN 4000 POSITION 9,9:PRINT "
       NUMBER OF PLAYERS (1
       /2)?";
FF 4010 GET #1,NP:IF NP<>49
       AND NP<>50 THEN 4010
KG 4020 NP=NP-49:IF NP=0 THE
       N 4060
IM 4030 POSITION 9,11:PRINT
       "NUMBER OF JOYSTICKS
        (1/2)?";
FI 4040 GET #1,JS:IF JS<>49
       AND JS<>50 THEN 4040
NK 4050 JS=JS-49
KK 4060 RETURN
KJ 4500 PRINT "NUMBER OF MAT
       CHES (1-27)";:INPUT
       S$:IF S$="" THEN S$=
       "0"
HB 4510 NM=VAL(S$):IF NM<1 O
       R NM>27 THEN NM=27
KL 4520 RETURN
JK 10000 DATA 81,69,65,82,82
       ,68,90,67
NH 10010 DATA 209,207,0,74,7
       2,0,154,131
KO 10020 DATA 12,3,0,93,91,0
       ,3,86
DK 10030 DATA 2,86,75,79,75,
       79,2,86
GJ 10040 DATA 20,84,0,212,21
       2,0,84,20
JG 10050 DATA 96,128,0,96,96
       ,0,128,96
HG 10060 DATA 8,128,0,28,30,
       0,128,204
KI 10070 DATA 73,79,65,83,83
       ,68,75,76
EG 10080 DATA 201,207,26,84,
       84,26,203,204
IG 10090 DATA 11,128,66,3,3,
       86,128,15
GE 10100 DATA 209,195,75,82,
       82,79,209,195
MJ 10110 DATA 124,124,0,60,6
       0,0,124,124
EJ 10120 DATA 61,61,217,0,0,
       89,61,61
NH 10130 DATA 254,254,0,31,3
       1,0,254,254
FE 10140 DATA 109,109,45,55,
       45,55,119,119
MB 10150 DATA 1,4,65,68,65,6
       8,1,4
BO 10160 DATA 1,68,0,139,139
       ,0,1,68
DA 10170 DATA 128,128,0,211,
       211,0,203,204
NN 10180 DATA 128,128,0,72,7
       4,0,203,204
OE 10190 DATA 73,79,0,128,12
       8,0,203,204
IG 10200 DATA 73,128,0,90,69
       ,0,128,76
OG 10210 DATA 209,128,0,90,6
       9,0,128,209
DP 10220 DATA 128,128,0,251,
       251,0,128,128
BM 10230 DATA 71,70,0,66,86,
       0,70,71
GB 10240 DATA 213,213,0,0,0,
       0,206,206
OA 10250 DATA 6,7,0,0,0,0,7,
       6
PN 10260 DATA 198,198,0,70,7
       0,0,198,198
EK 10270 DATA 128,128,0,128,
       128,0,128,128
FF 20000 REM Call with PIC h
       olding the picture
       number, XPOS & YPOS
        holding the card p
       osition (0-8,0-5)
JL 20020 RESTORE PIC*10+1000
       0
JO 20040 START=SCREEN+YPOS*1
       20+XPOS+XPOS
CK 20050 FOR I=1 TO 2:READ A
       :POKE START+40+I,A:
       NEXT I
BK 20060 FOR I=1 TO 4:READ A
       :POKE START+61+I+XP
       OS+XPOS,A:NEXT I
FJ 20070 FOR I=1 TO 2:READ A
       :POKE START+100+I,A
       :NEXT I
NK 20080 RETURN
```

On Balance

James V. Trunzo

Requirements: Apple II-series computer with a minimum of 128K. Both 3½-inch and 5¼-inch disks contained in each package.

After a hard day's work, few people want to take time to manage the money they make. In our house, that task is ably handled by my wife. Now, in order to make her job a little easier, I had, in the past, tried to get her to use my computer and any number of home accounting software packages. My efforts, however, were in vain. Either it took too many hours to set up accounts prior to even beginning to use the software, or the programs were too complicated for those of us without a degree in accounting. All we really needed was a computerized checkbook program with the ability to do some neat things, like sorts. Nothing we found could overcome her preference towards her own calculator, pen, and paper system. That is, nothing could until Broderbund released its newest productivity package: *On Balance*.

On Balance is a money management program—which, incidentally, is not copy protected—that so closely simulates the most basic of non-computer household systems, it destroys the inhibitions many people have about using a computer to aid in financial management. The fact that it emulates the system most households are already using makes it unintimidating to novices, giving them the security they need to fairly evaluate the program.

Simple And Versatile

On Balance allows the user to begin working with it immediately. After ten minutes of initial setup, you can begin making full use of the program. While you can create all your accounts before making any entries if you so desire, *On Balance*, unlike many other similar financial packages, doesn't require that you do so. Accounts can be established on the fly. If entering the details of a check is your first transaction, you set up your checking account at that time. If your next entry is a deposit into a savings account, just answer the series of onscreen questions, and another account has come into existence. Simplicity is one of *On Balance*'s best features.

However, don't confuse simplicity with lack of sophistication or get the idea that this program is a watered-down version of a "real" home accounting program—*On Balance* is as complete a program as any of its type currently on the market. It's just that Broderbund's program allows you a tremendous amount of versatility, and it doesn't require that you use all of its features if you need only a few. And it certainly has enough features. For instance: *On Balance* will handle all standard record keeping, reconciliation tasks, budgets, check printing, and so forth. It will generate reports showing your net income and net worth as well as reports on individual accounts and transaction lists. It will even print graphs allowing you to compare a variety of financial information, like money spent versus money budgeted. And it does all these things in a speedy, clear, simple way—which is what really separates it from others of its ilk.

Part of the ease of use implied above stems from *On Balance*'s use of pull-down windows and constant onscreen menus. Using a mouse, a keyboard, or a combination of both, the user selects major elements of the program from a menu line at the top of the screen. This drops a window containing more detailed choices. For example, selecting "Accounts" from the menu line will open a window listing all the account types that have been created. Furthermore, onscreen help is almost always available, and editing is a snap if modification is needed due to change or error.

Other features aid in the speed and usefulness of the program. An ever-present, full-functioning calculator hides behind the main screen, waiting to be summoned through the use of Open Apple-C, and a Notepad is also always available to jot down important notations about a transaction. Another feature that saves the user time and trouble is the ability to enter a set of regular payments, like a mortgage payment. For example, not only will *On Balance* prepare itself to handle these recurring transactions, it will also prompt you, through the use of onscreen messages, when one is due. Of course, these are just a few of the features built into *On Balance*.

Manipulating Data

Once you have entered information, you can manipulate it in a variety of ways. Searches and sorts can provide you with various forms of comparisons that will clearly show you where your money is—or where it went. Search by date, check number, payee, dollar amount, or even text; and send the data to a customized report. Then view the information that you requested either on your monitor screen or on paper. *On Balance* lets you record up to 800 transactions a month, and Broderbund claims that users can store an entire year's data on a single disk. This is an important factor if you wish to flag certain transactions throughout the year (for example, tax-related entries) and pull them all at one time.

One other thing: *On Balance* is compatible with *AppleWorks*. This means that the user can both export financial data from *On Balance* to *AppleWorks* and do forecasting and analyzing without having to re-enter start-up figures. A special disk facilitates the exporting of information from one program to the other.

Overall, *On Balance* is an impressive piece of work. It is designed to put you in charge of your financial affairs by giving you a wide variety of information in a simple, easily understandable fashion. Whether you're setting up a budget, keeping track of stock purchases, or simply balancing your checkbook, *On Balance* helps you do the job quickly and painlessly.

On Balance
Broderbund Software
17 Paul Dr.
San Rafael, CA 94903-2101
$99.95
(Interactive demo disk available for $7.99. Price can be applied to purchase of full package at later date.)

Amnesia

James V. Trunzo

Requirements: Apple II series, IBM PC and compatibles, and Commodore 64 computers.

It's not the type of thing one often, if ever, thinks about. Yet, for a moment, consider the terrifying prospect of complete memory loss. Your entire life disappears. All those you have grown to know and love—or even hate—become strangers. The career you've built and the knowledge you've gained dissolve into nothingness. Your solitude is so complete that, upon looking in a mirror, you don't even know the face staring back at you.

The rather unpleasant concept of complete memory loss, or amnesia, is the basis for Electronic Arts' first journey into the text adventure genre. Employing the considerable skills of Thomas M. Disch, winner of the Campbell Award for best science fiction novel in 1980, EA thrusts the player into the hazy world of an amnesia victim who lives in New York City. As this character, you wake up in the fictional Sunderland Hotel suffering from acute amnesia, and are lacking any comforting physical resources like food, clothing, or money. As you stumble about trying to piece together information that will literally return your life to you, you discover that things could get worse. As the package copy notes: "A strange woman wants to marry you. A strange man is trying to kill you. The state of Texas wants you for murder...," and you still aren't sure who you are.

Beyond The Ordinary

Amnesia, like other well-designed text adventures, puts you in a predicament and challenges you, with the aid of an extensive and sophisticated parser, to figure out which piece goes where in the giant puzzle. However, *Amnesia* goes well beyond ordinary text adventures, many of which arrived after Infocom's shining successes.

Electronic Arts lives up to its reputation by building its electronic novel in Manhattan—all of Manhattan. There are four thousand individual and authentic locations on this eclectic urban island, including 650 streets and the entire Manhattan subway system. Central Park, Chinatown, Soho, Broadway and 42nd Street, Times Square, Greenwich Village, the Battery, and even the Brooklyn Bridge are all faithfully reproduced. If you care to find out how faithfully, cruise Central Park at 2:00 in the morning (from the safety of your home). *Amnesia's* internal clock keeps very accurate time, so the muggers know when to come out. Actually, part of the enjoyment of playing *Amnesia* is wandering about fabulous Manhattan, especially if you're familiar with some of it. Even if you aren't, don't despair: Electronic Arts provides you with a detailed street map of the borough, as well as a complete map of the subway system.

Beyond the vast scope of the gaming environment that makes up *Amnesia*, Disch and EA have not glossed over the details that make day-to-day existence possible. In *Amnesia*, you need money or credit cards to function. Restaurants and stores open and close according to schedule. The television news (worth watching, incidentally) comes on at 7:00 in the evening. Your character is aware of the time of day, and it becomes hungry and sleepy.

Because *Amnesia* occurs in a modern-day setting that incorporates many familiar physical surroundings, it may appeal to some game players that don't enjoy the mythical or space-age formats that many computer games employ. This 1980's urban backdrop also provides the game with a sense of realism that may be a bit unsettling.

The program itself contains features typically found in text adventures. In addition to its excellent parser, *Amnesia* allows multiple saves, printouts, and a scoring system that rewards accomplishments beyond simply solving a piece of the puzzle.

The powerful combination of Thomas Disch's fine prose and EA's program design talent makes *Amnesia* a text adventure well worth experiencing.

Amnesia
Electronic Arts
1820 Gateway Dr.
San Mateo, CA 94404
$44.95 Apple II, IBM PC and compatible versions
$39.95 Commodore 64 version

Starglider

Andy Eddy

Requirements: Commodore 64, Amiga, Atari ST, Apple II, and IBM PC and compatibles. Disk drive required. Mouse optional, but recommended on systems where it is available; joystick optional, but recommended on all others. Color display also optional, but recommended. Atari ST and Commodore 64 versions reviewed.

Every once in a while, a game comes along that tickles and teases your senses. Remember the exhilaration you experienced the first time you played *Space Invaders* or *Asteroids*—the panic that overcame you as you got used to the controls, the racing heartbeat that seemed to match the pounding pulse of the sound effects, the adrenaline rush with every laser blast for or against you.

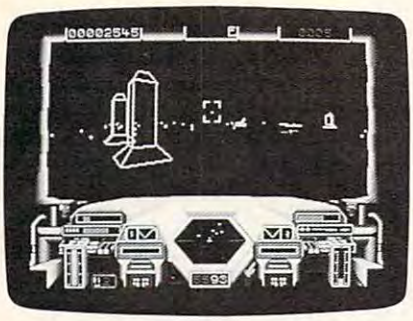
Over the years there have been some computer contests that touch you like this. One such product is *Starglider* by Rainbird Software (marketed in the U.S. by Firebird Licensees). A space sortie in vector graphics that can be likened to an aerial *Battlezone*, *Starglider* has a mix of bright colors, rapid movement, and strategic excitement that will bring you back again and again.

Find Out For Yourself

One of the most enticing points of the game is the inherent mystery that its creator has engineered. Much like a suspense novel, *Starglider's* charms aren't spelled out at the beginning for you. Given very little in the way of instruction beyond the most cursory navigational direction, your piloting skill is increased only through practice—during which time you will face many frustrating defeats—and careful perusing of the accompanying novella that details, in story form, the reason you are battling Herman Krudd and his troops. If your bent is to plunge into a game without reading through the manual, you'll find yourself in *deep trouble* time and time again.

Once you've acquired the knowledge to stay alive and keep your craft aloft (refueling your ship, replenishing your missiles, lasers, shields, and so on), and you've learned how to track all of the various indicators that alert you to your ship's condition, position, and endangerment, all you have to do is dispatch the many enemies that threaten you. Oh, is *that* all?

Most of these adversaries can be blown away with your shipboard lasers, though it takes quite a few shots for them to meet their demise. Others are impervious to your blasters and



The Atari ST version of Starglider *from Firebird.*

must be taken out with a missile—and your ship can hold only two at any given time. If this gives you the impression that your work is cut out for you, you're right.

Importantly, it must be pointed out that the discouragement level is not enough for you to shelve the game. Waving a galactic carrot in your windshield, *Starglider* teases you into the just-one-more-try mode. Most times, you find yourself getting a few more points on the scoreboard than the last time, maybe reaching a higher level than before. But with each level come new challenges that must be overcome. And on and on....

Jeremy "Jez" San of Argonaut Software (Jez San of Argonaut, Jason and the Argonauts—get the connection?), who originated *Starglider* and programmed the 68000 versions, is a talented individual who has constructed a visually smooth concoction that's so realistic in its feel that you'll duck and squirm in your seat in an attempt to guide your careening jet with body English.

A Loss In Translation?

Starglider was obviously designed on a 16-bit machine and converted to the 8-bit counterparts, and it suffers a bit in the transition. For example, the game is *very* well suited to mouse use, as I discovered in the ST version, for controlling altitude, speed, steering, and laser triggering.

But, in the Commodore rendering, the joystick seems to come up a little short in giving you carefree handling of the ship, though some latitude is provided by having two different joystick modes. Here, the keyboard assists on certain functions; in fact, all versions of the game can be played entirely from the keyboard, and the ability to redefine keys lets the player make a personal layout for game control.

Graphics differences are more drastically exposed. On the ST, a radar screen at the bottom of your dashboard shows all objects within a certain perimeter with each item's "blip" a distinct color. On the 64, it's impossible to distinguish one blip from the next, which puts you at a marked disadvantage when you're on the prowl for a docking silo or energy tower at critical moments. Other than that difference and a few variations in sound and visual effects between the two, the game play is very similar, and the differences only limit its spectacularity on the Commodore. I must say, the digitized voice status reports during play and the rockin' intermission music *with vocals* on the ST rendering were real shocks.

About the only across-the-board complaint I had was tolerable, to say the least. At the end of each foray, a high score table is displayed where you may insert your name among the top achievers—standard fare for most games of this ilk, and, above all, making for some heated competition between contestants. The problem is that the list lasts only the length of that immediate session; no scores are saved to disk for permanent recall. While San told me that this was to prevent the possibility of overwriting vital program data, in the past it has been a feature on many games with very little, if any, detriment.

To The Future

So far, Rainbird has brought *The Pawn* and *Starglider* to the Americas from overseas—and these are two of the most critically acclaimed programs in recent memory. If they keep this streak going, they can be counted on to become one of the major suppliers of quality gameware.

Starglider
Firebird Licensees
P.O. Box 49
Ramsey, NJ 07446
$39.95 Commodore 64 version
$44.95 all other versions

Robot Rascals

Karen G. McCullough

Requirements: Apple II series, Commodore 64.

Robot Rascals is a hybrid; it's a cross between a card game and a computer adventure—Go Fish meets the electronic scavenger hunt. In games, as in plants and animals, cross-fertilization has the potential to produce hopeless disasters as well as unusually strong, effective offspring. *Robot Rascals* is among the successes.

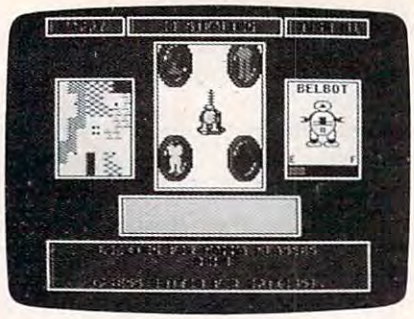
Each player (up to four) is dealt four item cards; then each selects a robot from the ten available onscreen. During a turn the player directs the robot on a scavenger hunt, looking for the items that match the cards in his or her hand. The player who gets to home first with items to match all the cards in his or her hand wins.

Sounds simple, right? You bet. Except for a few complications, like the luck cards you draw before each turn. These can direct you to take a card, steal a card, force a swap among other players, and so on, all of which can wreak havoc with your (or your opponent's) hand. The item deck contains a few surprises also: wild cards, a killer card, and a cosmic cheat, though these are used only in more advanced games. Then there's the problem of thieving robots, and, just to keep things from getting boring, the computer will occasionally change the rules.

If this sounds like overkill, take heart: The game can be played on several levels. Not all the complications apply at the lower echelons. To start, you can play a super-beginner game, a simple race to find the four items in your hand and beat your opponent(s) home. It's a good introduction, but only three- and four-year-olds won't be quickly bored and ready to move onto higher levels. The advanced game is a no-holds-barred free-for-all, with more complications than a jet fighter's controls.

This is a well-designed, well-executed game program. The screen windows give all the status information you need, and they show what your robot is up to. Joystick control is tight and precise, a real pleasure. There's plenty of variety in the terrain you search, and enough travel and movement options: Your robot can walk or teleport to get around, drop an item, or erect defensive shields.

But the real joy of the game is the robots themselves. There are ten to choose from, and each is distinctive; in fact, it's no exaggeration to say they



The Commodore 64 version of Robot Rascals *from Electronic Arts.*

have individual personalities. The animation of these little technological marvels is subtle and clever. Each robot moves differently; some grin while they walk. Belbot pounds his chest in delight when he finds an item; Birt jumps for joy. Sphero flops along lazily when he moves, but if you take too long to give him directions, he'll stamp his "foot" impatiently.

The whole family, including the three-year-olds, can play *Robot Rascals* since a handicapping feature lets players of differing ability compete against each other. If the game has a flaw, it's that it's—pardon the pun—almost terminally cute. But then, so is Teddy Ruxpin, and we know how many of those have sold.

Robot Rascals
Electronic Arts
1820 Gateway Dr.
San Mateo, CA 94404
$39.95 Commodore 64 version
$44.95 Apple II version

Jet

Michael B. Williams

Requirements: Apple II series with 64K minimum, Commodore 64, or IBM PC/PCjr and compatibles with 128K minimum and color graphics adapter. Joystick optional. Apple version reviewed.

It's the realtime, three-dimensional display that best exemplifies this SubLOGIC program. Everything—from planes, missiles, and mountains, to the runway and control tower—is shown in perfect perspective and color. If you were to fire a missile at a plane passing in front of you, you would see not only the missile eject from your jet, but also, in perspective, the unwary plane approach from the side, be hit by the missile, and break into pieces as it falls toward the ground.

Jet has several display enhancements to help you maneuver the aircraft. As if using a telescope, you can zoom in to see objects in greater detail and zoom out to expand your field of vision. The attitude (pitch) indicator, which can be superimposed over the display, rotates, rises, and falls to reflect your orientation to the horizon. It, too, is seen in perspective, with its ten-degree graduations becoming smaller towards the center of vision. To help you in battle, you can summon a color-coded radar display that shows the location of enemy planes and missiles and of your home base—all with respect to your aircraft. *Jet* can also supply a range indicator that changes color as you close in on a target.

Jet's instrument panel is sparse compared with those of other flight simulators. It is this deceptively simple display that allows you to ignore many of the technical aspects of flying and to concentrate on the flying experience itself. At the same time, this makes *Jet* noticeably less realistic than SubLOGIC's *Flight Simulator II*.

Flying By Remote Control

In addition to the normal cockpit display, there is a unique feature which allows you to pilot the plane from the control tower. In essence, you are flying the plane by remote control. Your field of vision is fixed toward the aircraft, and you can see the plane as it is taking off and landing. Since the only display feature that is accessible in this mode is *zoom*, you will find it difficult to fly the jet this way. You can easily toggle back to the cockpit display at any time. There, in addition to the forward view from the cockpit, you can also see above and to the left, right, and rear of the plane.

Jet does have a few problems which surface because of the speed and graphics limitations of the Apple II: Sound effects are sparse. The program is painfully slow at updating the display, which can turn a smooth flight into a spasmodic one. The program is slowest in its combat mode; screen updates occur at about one second intervals. Most importantly, you *must* use a color monitor with the Apple version of *Jet*. On a monochrome monitor, it is nearly impossible to distinguish between the sky, horizon, and markings on the ground; attempting to land the jet becomes a daredevil event at best. With a color monitor, however, each element has a unique color to help distinguish it.

A few of *Jet*'s problems are due to its implementation. The aircraft's speed is given as a Mach number (relative to the speed of sound) on a graduated scale which is only marked in increments of Mach 0.5. The altitude is also represented by a vertical graduated scale, but is equally difficult to read because it is marked with increasing intervals instead of constant ones. Digital readouts for both the speed and the altitude would help tremendously. When you are fighting MiGs, you won't have time to guess your actual altitude and speed.

To add to the feeling that you are really flying, *Jet* includes real hazards such as blackout and red out, which reflect the human body's limited tolerance to high acceleration. In the event of imminent destruction, you can push the eject button and float safely back to earth by parachute.

Jet comes with a quick reference card summarizing the commands available. It is also compatible with the same scenery disks used by *Flight Simulator II* and Microsoft's *Flight Simulator*. SubLOGIC offers scenery disks for Japan and the San Francisco area.

If you have a color monitor, you'll definitely want to consider adding the state-of-the-art *Jet* to your program library. Its lack of realism may turn off flying aficionados, but its ease of flight and its truly remarkable graphics are sure to please weekend pilots who want to take their F-16 for a spin around the Golden Gate Bridge.

Jet
SubLOGIC
713 Edgebrook Dr.
Champaign, IL 61820
$39.95 Apple II series and Commodore 64 versions
$49.95 IBM PC/PCjr (and compatibles) version

Computers and Society

David D. Thornburg, Associate Editor

A Look At An Era

When there's talk about the success stories of the personal computer industry, most people think of Silicon Valley with all its ups and downs. Names like Jobs, Wozniak, Peddle, and others too numerous to mention, are bandied about as though they were the only people involved with the success of this industry.

While no one would want to diminish the contributions of these people, the fact remains that there is another part of the personal computer industry located far from Silicon Valley with success stories of its own. As valuable as your personal computer might be, you probably wouldn't know more than half the things you can do with your computer if it weren't for magazines like COMPUTE!.

Now this piece is not a pitch to get you to read COMPUTE!—after all, you are doing that already. The reason for spending time on this topic this month is that, with the departure of Robert Lock from the day-to-day operations of this magazine, COMPUTE! has entered a new era. Accordingly, I thought I might share some of my recollections on the growth of this magazine since I had an article in its very first issue, and have had at least one article in nearly every issue since that one.

Retrospectives of this sort are usually to be found in the last issue of a magazine just as it goes belly up. As you know, many computing magazines have fallen prey to the vagaries of the computer market. Some of the older magazines (like *Recreational Computing*) were acquired by other magazines (like COMPUTE!), and still others just quietly went out of business (like *Creative Computing*).

COMPUTE! has had its ups and downs—just as has the industry—but unlike most of its counterparts, COMPUTE! has emerged stronger than ever for one simple reason—its readers.

A dedicated base of readers is essential to the survival of a magazine, and COMPUTE! was careful from the start to insure that it had a solid base of editorial writers who helped maintain the consistency that made this magazine what it is.

In the fall of 1979 the first issue of COMPUTE. The Journal of Progressive Computing hit the stands. It was 104 pages long and contained 19 articles, 10 reviews, and a full spectrum of advertisers from Commodore to small garage operations. My company, Innovision, placed its first ad in this premier issue. While my company hasn't grown much in the intervening years, it is still around.

While the major focus of COMPUTE! was on the Commodore PET, it also devoted space to other 6502-based computers like the Ohio Scientific Challenger (remember that one?) and the single-board computers like the KIM-1 and AIM-65 (may they rest in peace).

While COMPUTE! started as a quarterly, Robert decided to make it bimonthly starting with the January/February 1980 issue. By this time the magazine was publishing articles about the Apple II and Atari 800, as well as continuing its strong PET coverage.

By the third issue, this column was started. This means that COMPUTE! has had the longest-running column on the social impact of computers in the history of personal computer magazines.

Within a short time COMPUTE! became a fancier magazine, sporting a full-color cover and monthly publication. Our magazine had come of age.

ABC

I was concerned when ABC acquired the magazine. I was afraid that COMPUTE! was going to lose some of its personal touch. But, under Robert's careful guidance, this never happened. Even though I am on the West Coast and have never visited COMPUTE!'s offices, I could tell from my phone conversations that Robert was hiring exactly the right kind of people to let the publication grow and thrive.

From the humble beginnings of COMPUTE!, Robert built a multifaceted publishing venture that included several magazines and a full catalog of books. Furthermore, he did this during a time when the computer industry was on a roller-coaster ride of immense proportions.

The Lesson?

When the computer magazines started to drop like flies, the "smart money" people said the the survivors would be the highly focused one-machine magazines. Magazines with a general focus were going to be victims simply because advertisers would not be able to target their ads as carefully.

Because COMPUTE! also published machine-specific magazines (like GAZETTE for Commodore owners), it could offer advertisers what they needed—and this probably helped maintain the magazine's success.

Can a general personal computing magazine survive? COMPUTE! has shown that it can not only survive, but that it can, with your support, thrive in both good times and bad.

The main reason, as I said before, is because of you—the readers of this magazine.

Another reason—one I think is equally as important—is because this magazine was built with the leadership of one of the finest men I have met in the industry.

Thank you, Robert, for all you have done for all of us. Our entire industry is watching to see what you will be doing next. ©

Microscope

Sheldon Leemon

Sometimes the impact of a rumor is more significant than whether or not it proves, in the end, to be true. A good example is the new generation of personal computers that IBM may or may not have announced by the time that you read this. For months, we've been hearing about as many as three different PCs that IBM may come up with, with code names like "Renegade" and "CloneKiller." The most often discussed is a low-end machine for the home and educational market. The machine described doesn't have expansion slots—at least not any compatible with current models—although it's said to have some networking capabilities (vital in an education setting). It will likely use the 80286 processor running at a slow six megahertz, and have a built-in graphics adapter, but rumors here range from a normal EGA-type adapter to one with fabulous graphics capabilities, such as 640 × 480 resolution with up to 256 colors onscreen at once, and windowing capabilities in ROM. Likewise, sound may be anything from an internal beeper to a full-blown synthesizer. All rumors agree on a 3½-inch disk drive and a full keyboard.

The operating system is another area of dispute. Most sources agree that it will use a new DOS, but reports on its features range from a slight change to support networking, to a version that includes *Microsoft Windows* or *Topview*, to a hybrid MS-DOS with a proprietary hardware/software scheme to shut out would-be cloners, to a completely proprietary system. In all of the rumors, we hear again and again of the possibility of a completely new hardware bus that will not accept the thousands of third-party add-on products for the current PC, and of a completely new operating system that will be upwardly compatible with the current one, but that can't be copied. In short, the anticipated PC wouldn't be PC compatible, and couldn't be cloned.

At first, we heard that the machine would be announced during the Super Bowl, à la Apple. When that didn't come to pass, stories began to center around a big meeting of IBM dealers in March, and a possible April announcement. But whether or not any of the rumors turn out to be true, their existence is being felt in the PC marketplace. Corporate buyers are holding off, waiting to see what develops before they commit to new purchases. The mood is reminiscent of that which prevailed before the announcement of the PCjr, when a "PC II" was rumored to be imminent.

But rather than being the product of a conspiracy on IBM's part, as some have suggested, it may well be that these rumors are a reflection of a growing perception of the seriousness of IBM's position. Having spawned the enormous PC market, IBM has had to stand idly by and watch its influence in that market diminish. Whether IBM will take bold action, and whether that action will be enough to stem the tide, are questions that will keep everybody in the industry watching.

****

Although the new high-end Macintosh models discussed in this issue's "Editor's Notes" are priced beyond the means of the average consumer, they're bound to have an effect on the home computer market, just as the original Mac did. For one thing, their introduction will send down the price of the Mac Plus and 512K Mac. The street price of the latter may break the $1,000 barrier for the first time, putting it head to head with the Atari ST, the Amiga, and PC clones.

The small screen and closed architecture of the old Mac are much less of a problem with the home market than with the business crowd, and users may be more apt to put up with these limitations for the time being, knowing that upgrades are available. Since software and the time spent creating data files usually end up as the biggest part of a computer investment, it's quite important that the user know that when he buys new hardware, he'll be able to take his software with him. Atari and Commodore have stated their intentions to manufacture 68020 machines, but so far, only Apple has assured a compatible upgrade path.

****

Although the Atari PC clones caused quite a sensation at CES, there have been recent reports that they may not appear quite as quickly as expected. Though Atari claimed that the machines would be ready by March, it may be as late as August when they are actually sold. For one thing, they have yet to undergo the sometimes-lengthy FCC approval. For another, Atari has yet to sign an agreement with Digital Research allowing it to distribute the GEM operating system with the machine. Though one distributor was quoted as saying he had the machines in the warehouse, others have doubted that the models shown at CES were actually the finished product.

In the meantime, Atari's announcements may end up hurting ST sales, with buyers waiting for the new PC clones or Mega ST machines. And though Atari cut the price of the current STs at the time the new machines were announced, dealers whose stock was purchased at the old price may be reluctant to sell them at as large a discount as some consumers will expect. ©

Telecomputing Today

Arlan R. Levitan

Twelve Special Bulletin Boards

While most continue to be microcomputer related, a growing number of electronic bulletin board systems (BBSs) have veered off the beaten track. BBSs devoted to law, medicine, genealogy, and real estate are common enough to elicit little more than a yawn from seasoned telecomputerists. Here are a dozen free boards that will spice up even the most jaded palate. Bon appétit!

Note: All numbers were verified as of February 20, 1987. Please observe board rules and common courtesy. Remember, you are a guest in the system operator's (SYSOP's) "house."

Aviation Connection
(214) 245-5633
Dallas, TX

You don't need a 2400-bps modem to fly around here. Whether you're certified for flying on instruments or just an aeronautics buff, the Aviation Connection is dedicated to your wild blue yonder.

Bullet 'N Board
(703) 971-4491
Silver Spring, VA

SYSOP Tanya Metaksa's aim was to dedicate this board to the Second Amendment and firearms. News on the latest legislative happenings and weaponry. Gun-show schedules and National Rifle Association information abound. While this board is free, you must go through a registration process to gain access.

The Casino BBS
(609) 652-6030
Atlantic City, NJ

Feel lucky? You won't lose your shirt playing in this casino. SYSOP Dave covers the Atlantic City casino beat, including nightlife and entertainment guides. Ask regulars where the best slot payoffs are and how much it takes to build a hotel on Boardwalk these days.

Collectors Network
(213) 204-0646
Los Angeles, CA

Just how much is that Charlie "Sunday Punch" Maxwell card worth? SYSOP Harry Rosenfeld knows. Info on coins, stamps, baseball cards, and just about anything else that's collectible. Also includes excellent BBS lists.

Crime Prevention BBS
(214) 578-1311
Plano, TX

Who broke Emma's window last Thursday night? Follow the saga of crime in Plano, Texas. Tips on spotting con artists, prevention of criminal mischief, and personal protection—all from SYSOP Captain Lyndon Payne and the rest of Plano's finest. Be sure to check out the "Crime of the Week."

Cryptologic Research
(703) 237-4322
McLean, WV
Hours: 5:30 p.m.–8:00 a.m. EST M–F

Do you suspect that the scribblings of your three-year old are really cleverly coded messages for special agents? SYSOP Robert Juneman operates this board as a service to the International Association for Cryptologic Research (IACR) and anyone else interested in Cryptography and Computer Security.

Electronic Call Board
(718) 499-1633
Brooklyn, NY

Dedicated to the performing arts, SYSOP Bobby Ballard keeps aspiring actors apprised of the latest casting notices. Special-interest sections covering theater, film, video, music, and art. If the Muse moves you, participate in electronic role playing. The Call Board also includes schedules of stage shows playing around the country.

The Guideboard
(415) 864-3858
San Francisco, CA

Get a real hacker's view of one of America's most popular vacation spots. The Guideboard is frequented by cabbies who keep each other posted on what's going on in the city by the Bay. Enough colorful personalities to populate a season's worth of "Taxi" TV reruns.

MIDI World Network
(213) 826-4288
Los Angeles, CA

SYSOPs Moore, Daystrom, and Fitzpatrick are in tune with the times. An excellent BBS devoted to MIDI-related computer use. Highly recommended if one of your keyboards has black and white keys.

Survival Communication
(707) 545-0746
Napa Valley, CA

Pack the freeze-dried food, hop in the jeep, and head for the mountains. Don't forget your modem-equipped lap-top, though; there are forums on survival, self-sufficiency, and emergency preparedness. SYSOP Don Kulha hosts discussion areas on medicine, food, alternative energy, radio communications, weapons-craft, and survival vehicles.

Top of the Rockies BBS
(303) 963-3688
Roaring Fork, CO

Is warmer weather tempting you to hang up the skis and poles for the season? Let SYSOP Barry Clements tempt you with ski information for Aspen, Snowmass, Sunlight, and the rest of the country. If you get tired of discussing equipment and technique, check out the tasty recipes and nutrition information.

The Train Board
(513) 398-0928
Mason, OH

Does the thought of the electric "chug" of an ancient Lionel train set running in your living room bring tears to your eyes? Or do you prefer using radio-controlled submarines in the local duck pond? SYSOP Decker Dogget moderates information on train collecting and radio-control hobbies. ©

AmigaView

Sheldon Leemon

Hardware Add-Ons

There are an astonishing number of good hardware products for the Amiga 1000, and more are waiting in the wings. The following sampling represents products that are currently available, or that should be by the time this column appears. I've chosen to list only hard disks, RAM expansions, and clock calendars, since these represent the categories in which users are most interested.

Hard Disks

Two outstanding products in the hard disk category are the MAS-20 from Microbotics, and the C Ltd. drive, both of which provide 20 megabytes of storage and a SCSI port for a list price of about $1,000. C Ltd. has lately added a full line of higher-capacity drives, ranging from 30 megabytes for $1,300 to 350 megabytes for $7,000. Xebec is a name that's new to the Amiga community, but it's very familiar in the IBM PC world, where the company is a leading maker of hard drives and controllers. Just out is the Xebec 9720H 20-Meg SCSI hard drive, with a list price of $1,075. I've used a preliminary unit, and my timings indicate that this drive loads files a bit faster than the C Ltd. and Microbotics drives.

Two new drives should be available for sale by the time you read this. The first is from Supra Corporation, which has an established track record with hard drives for the Atari ST and the Macintosh. The Supra 20-meg hard drive lists for $1,000, and offers an optional 1-meg-RAM upgrade board that fits into the drive controller card. The second is the PAL Jr. from Byte by Byte. Having gone through many design changes, the final version will be a mini version of the PAL box, an expansion box that fits on top of the Amiga. Though only 2½ inches tall, it will come with a 20-meg hard drive and a meg of RAM, and have two full-size horizontal expansion slots. The price remains at $1,500, and because all sales will be direct, there will be no discounts from list price. For this price premium, Byte by Byte hopes to offer much higher performance.

RAM Expansion

RAM expansion units for the Amiga have really proliferated lately. Prices keep changing so quickly that it would be pointless to give exact costs here, but at this time, one-meg boards range from $300 to $450, and two-meg boards from $575 to $850. Many of these are in the form of self-contained modules that plug into the right side of the Amiga. In this category there's the the one-meg aMEGA from C Ltd.; the Xpander II from Pacific Cypress, a two-slot box that comes with a two-meg card in one slot; and my own favorite, the Starboard 2 from Microbotics. The Starboard is a compact unit which holds from 512K to two meg, and has provisions for a multifunction card with clock calendar and 68881 math coprocessor.

The Insider, from Michigan Software Distributers, is a new one-meg board that mounts internally. It plugs into the 68000 processor socket, and includes a clock calendar. Another internal expansion is the Kickstart Eliminator and RAM Expansion Kit from Creative Microsystems. This isn't strictly a RAM expansion, since the kit provides the Kickstart 1.2 code on EPROM chips. Not only does this eliminate the need for the Kickstart disk, but it also frees up the 256K of write-protected RAM for general use. A couple of caveats apply. Installation is not for the inexperienced, and it voids your Amiga warranty. With Kickstart in ROM, you can't switch versions without changing chips, which means you can't run software that only works with 1.1. But Sidecar or hard disk users, who will want to use 1.2 exclusively, won't find anything better for convenience and extra memory.

ASDG also makes memory expansion boards that fit in expansion boxes like the PAL Jr., and its own Mini-rack. But its most exciting product may well be the Recoverable RAM Disk, a shareware program which creates a RAM drive that survives a warm reset (CTRL-Amiga-Amiga). It's available on most of the information services and bulletin boards.

Clock Calendars

There's a clock-calendar board available for almost every port on the the Amiga. Tic from Byte by Byte and MouseTime from Microbotics connect to the second mouse port. MouseTime fits next to memory expansion modules, but Tic doesn't, necessitating software that switches the function of the two mouse ports. Atime, from Akron Systems, sits on the printer port, and provides a pass-through for the printer. The most innovative, though, may be Time Saver from C Ltd., which connects in the keyboard line. Not only does it update the system time automatically at power-up time, without software, but it also has 8K of battery-backed-up RAM for keyboard macros and CLI command history. ©

Train for the Fastest Growing Job Skill in America

Only NRI teaches you to service all computers as you build your own fully IBM-compatible microcomputer

NEW! Train with the newest Sanyo 880 Series Computer – it's fully IBM-compatible and runs almost twice as fast as the IBM PC!

With computers firmly established in offices—and more and more new applications being developed for every facet of business—the demand for trained computer service technicians surges forward. The Department of Labor estimates that computer service jobs will actually *double* in the next ten years—a faster growth rate than for any other occupation.

Total systems training

No computer stands alone... it's part of a total system. And if you want to learn to service and repair computers, you have to understand computer *systems.* Only NRI includes a powerful computer system as part of your training, centered around the new, fully IBM-compatible Sanyo 880 Series computer.

As part of your training, you'll build this highly rated, 16-bit IBM-compatible computer system. You'll assemble Sanyo's "intelligent" keyboard, install the power supply and disk drive and interface the high-resolution monitor. The 880 Computer has two operating speeds: Standard IBM speed of 4.77 MHz and a remarkable turbo speed of 8 MHz. It's confidence-building, real-world experience that includes training in programming, circuit design and peripheral maintenance.

No experience necessary— NRI builds it in

Even if you've never had any previous training in electronics, you can succeed with NRI training. You'll start with the basics, then rapidly build on them to master such concepts as digital logic, microprocessor design, and computer memory. You'll build and test advanced electronic circuits using the exclusive NRI Discovery Lab®, professional digital multimeter, and logic probe. Like your computer, they're all yours to keep as part of your training. You even get some of the most popular software, including WordStar, CalcStar, GW Basic and MS DOS.

Your NRI total systems training includes all of this:
NRI Discovery Lab® to design and modify circuits • Four-function digital multimeter with walk-you-through instruction on audio tape • Digital logic probe for visual examination of computer circuits • Sanyo 880 Series Computer with "intelligent" keyboard and 360K double-density, double-sided disk drive • High resolution monochrome monitor • 8K ROM, 256K RAM • Bundled software including GW BASIC, MS DOS, WordStar, CalcStar • Reference manuals, schematics, and bite-sized lessons.

Send for 100-page free catalog

Send the post-paid reply card today for NRI's 100-page, full-color catalog, with all the facts about at-home computer training. Read detailed descriptions of each lesson, each experiment you perform. See each piece of hands-on equipment you'll work with and keep. And check out NRI training in other high-tech fields such as Robotics, Data Communications, TV/Audio/Video Servicing, and more.

If the card has been used, write to NRI Schools, 3939 Wisconsin Ave., N.W., Washington, D.C. 20016.

NRI is the only technical school that trains you as you assemble a top-brand microcomputer. After building your own logic probe, you'll assemble the "intelligent" keyboard...

...then install the computer power supply, checking all the circuits and connections with NRI's Digital Multimeter. From there, you'll move on to install the disk drive and monitor.

NRI SCHOOLS
McGraw-Hill Continuing Education Center
3939 Wisconsin Avenue, NW
Washington, DC 20016
We'll Give You Tomorrow.

IBM is a Registered Trademark of International Business Machine Corporation.

The Beginner's Page

C. Regena

Sound And Music In BASIC

Programming sound and music on your computer can be both fun and rewarding. It's among my favorite things to do on the computer. I wish I could just give you a BASIC listing for some music on your specific computer this month, but that's not possible, since this is a column for all computers, and music commands in BASIC are very machine-specific. However, here are some general ideas you can use when programming sound and music.

The Most Common Commands

To program your own music or sounds, you'll need to refer to your BASIC programming manual. The most common key BASIC words for sound are PLAY, SOUND, BEEP, and WAVE. Look these words up in your index or list of BASIC words to see if they are available on your computer. The Commodore 64 uses POKE commands, so refer to the chapter on sound in your manual as well as the pages in the appendices that list the memory locations and values for sound, volume, and voice programming.

On some computers you may be able to use the command **PRINT CHR$(7)**, which is the bell or beep character for a short tone.

The PLAY command usually uses note names in quotes to "play" musical tones ABCDEFG. You may also specify sharps with # or + and flats with −. To specify which octave for the note you want, use the letter O before an octave number, such as O3. Other options include the length, L; a pause (rest), P; a dot for dotted notes; and tempo, T. The IBM PLAY statement has many more options which are listed in the manual. A sample command is

```
30 PLAY "O3 CBABCDE"
```

After you get used to the general PLAY command, try using string variables to play a longer tune or to play repetitious phrases without typing a lot of individual PLAY statements. For example, let A$ be the string to play one musical phrase. Then you can use the command PLAY "XA$;". You may use numeric variables also. For example, if you have a variable octave J, use the command PLAY "O = J;". Consult your manual for the use of variables and to determine when the semicolon is necessary.

Different computers have different variations of the SOUND command. Here are some examples:

IBM: SOUND *frequency, duration, volume, voice*
Atari: SOUND *voice, note, tone, loudness*
Atari ST: SOUND *voice, volume, note, octave, duration*
Amiga: SOUND *frequency, duration, volume, voice*
Commodore 128: SOUND *voice, frequency, duration*

Any of the parameters may be numeric variables, but you do need to make sure the variables are within the proper limits.

For the Atari and Atari ST commands, the notes are numbered. You can refer to charts to see the numbers that correspond to each musical note and tone or octave.

Sound Frequency

Some SOUND commands use a *frequency* parameter. The frequency is the cycles per second, or Hertz (Hz), that produce a particular tone. For example, concert *A* is 440 cycles per second. Your manual should have a chart comparing note names to frequencies. You might keep in mind that a note one octave higher is double the frequency. For example, concert *A* is 440Hz, and the *A* note one octave higher is 880Hz. The note *A* one octave lower is 220Hz.

The *duration* parameter is a number that tells how long to play a note. On the IBM and Amiga, the duration is figured in "clock ticks," which occur 18.2 times per second; on the Atari ST the duration is the time in 1/50 second counts; on the Commodore 128 it's in 1/60 second counts.

On most computers, the volume parameter is a number from 0 to 15, with 15 the loudest. The Amiga uses numbers from 0 to 255. The voice parameter refers to the sound channel you want to use. Using different voice numbers allows you to play more than one note at a time, as when playing a three-note chord.

Other commands may be associated with the SOUND command. You may need to use a delay loop instead of specifying a duration (Atari). If so, you could use a FOR-NEXT loop in a subroutine. In IBM BASIC, be sure to read about MB for the differences in music background and music foreground.

The WAVE command for the Amiga and Atari ST computers are very versatile commands that change the waveforms of the sounds so you can get, for example, white noise, a flute sound, or a trumpet sound. (POKE commands on the Commodore 64 change waveforms.) Since numeric variable names are allowed in SOUND commands, try using variable notes in FOR-NEXT loops for all kinds of different sound effects. You might keep the note number or frequency the same, but vary the volume. Or try a FOR-NEXT loop with the frequency increasing in the loop.

You might want to experiment a bit with PLAY and SOUND to create your own computer musical composition. You may need to experiment a lot to get the sound you want, but the results can be very satisfying. ©

A Magic Slate For Young Writers

I am currently working with Alabama's Dr. Gloria Solomon and Canada's Dr. Julie Davis to develop multimedia presentations using computers in educational environments. For our word processor we have chosen *Magic Slate* from Sunburst Communications (39 Washington Ave., Pleasantville, NY 10570-9971). *Magic Slate* costs $99.95 and is available for the Apple II family of computers with a minimum of 48K. (An 80-column card is needed if you use the 80-column version.)

Magic Slate is a full-function word processor. It lets your young authors do all the basic word processor functions, including cutting and pasting, word-wrap, centering, underlining, search and replace, and so on. It comes in a large orange notebook with a backup disk, a teacher's guide, and primary and advanced student handbooks which the manufacturer encourages you to reproduce. For an extra price you can also get a lab pack and extra reference cards.

Electronic Pen Pals

We chose *Magic Slate* for three major reasons. First, it comes in 80-column, 40-column, and 20-column versions. Teachers and older students find the 80-column version easy to learn and use, yet comparable to "business"-quality word processors. Younger students delight in the 20- and 40-column versions. They especially like the 20-column *Magic Slate*'s big letters. It's easy to fill a screen with these letters, and you can print them out on paper if you have a graphics printer.

Second, *Magic Slate*'s utility function makes it easy to convert students' papers, reports, and stories into files which can be sent via modem to other students thousands of miles away. There is so much more incentive for students to write when they know their words will be transmitted quickly to other students over the telephone line. Hundreds of students have become electronic pen pals, and several students in Alabama and Canada are collaborating on research and science reports for their teachers. Another dozen students are jointly authoring an electronic novel which is presently growing at a rate of *five new chapters a day.*

Third, *Magic Slate* does not exist in a vacuum. It is supported by an excellent family of writing programs which enhance and extend the basic word processor. The first program is *Type to Learn* (for grade level 2–adult students, $69). *Type to Learn* teaches students how to use the computer keyboard. Since the program uses a language-based approach, students not only learn where the keys are on the keyboard; they also practice their spelling, composition, grammar, and punctuation as they type. (At extra cost teachers can purchase a ten-disk lab pack, student typing textbooks, and a gradebook disk to manage students' keyboard activities.)

Next come a group of three programs: *I Can Write!* ($40, for grade 2 students), *Be a Writer!* ($40, for grade 3 students), and *Write with Me!* ($59, for grade 4 students). Each program contains 25 lessons which take the student, step by step, toward becoming a young author. *I Can Write!*, the most elementary program, starts with open-ended writing exercises which encourage a student to explore his or her personal identity. *Be a Writer!* carries beginners into more formal language objectives, including the construction of full sentences, and using descriptive, narrative, and explanatory writing. The third program, *Write with Me!*, lets children construct their own book, 25 chapters long.

The Collected Writings

Students use the 20-column version of *Magic Slate* when they are doing *I Can Write!* and *Be a Writer!* activities; they use the 40-column version of *Magic Slate* with *Write with Me!* The programs challenge students to develop their word processing skills along with their language skills. As their writing ability increases, students are encouraged to use more advanced word processing functions. Teachers can use a printer to print out the students' compositions. Each page of a student's work adds to a growing book of his or her writings. After three years and 75 chapters, a student's "collected writings" can be quite impressive.

One last program, *Magic Slate Typestyles* ($49 for either the 20-column or 40-column *Magic Slate*), lets students install new typestyles on their *Magic Slate* disk. Students can use premade typestyles or design new typestyles of their own with the program's powerful editor. Teachers especially like the typestyles program because it enables them to teach students that learning to write no longer means just putting words on a page (or screen). Now a person who wishes to communicate can also be involved with the way the writing looks. With this program, the writer must choose a page's layout and design, the character set and font being used, and the accompanying graphics. Even for second- and third-graders, desktop publishing is right around the corner. The *Magic Slate* family is so valuable because it prepares youngsters for the age of desktop publishing by integrating language arts, word processing, and "page processing" skills into a single curriculum of exercises and activities.©

INSIGHT: Atari

Bill Wilkinson

RUN And INIT Vectors

This month's discussion is something of a continuation of my column of a couple of months ago, where I presented a program that showed you the segments of a binary file. And that column, in turn, referred back to the April 1986 column. Both columns are required reading for a full understanding this month, but you'll learn something even if you are reading this cold.

We begin by noting that when you ask Atari DOS (version 2.0S or 2.5) to save a chunk of memory as a binary file, it asks you to supply four numbers:

START,END,INIT,RUN

And, if you've looked through enough magazine articles or user group newsletters, you've probably come across places where an author instructed you to use the save binary file option, mentioned the beginning and ending addresses, and then told you to be sure to give the proper RUN (and/or INIT) address. The START and END numbers seem obvious: They are the first and last addresses of the range of memory to be written out. But what about INIT and RUN? What can those possibly mean?

A Feature Unmatched

The ability of *any* binary file, including the ever-important AUTORUN.SYS, to have a RUN or INIT address associated with it is, in my opinion, a feature unmatched by any small system DOS, up to and including MS-DOS (IBM PC and clones) and TOS (for the ST). Only with Atari DOS's binary files and their format-compatible relatives can you tell the operating system to load part of your binary file (also called machine language file, object code, and so on—several names for the same thing), execute that part, and then continue loading more of the file. So let's see how it all works.

When DOS loads a binary file, including the AUTORUN.SYS file at power-up time, it monitors two locations. The simpler of the two is the RUN vector. Before DOS begins the load of a binary file, it puts a known value into locations 736–737 (hex $2E0–$2E1). When the file is completely loaded—DOS encounters the end of the file—if the contents of location 736 have been changed, then DOS assumes the new contents specify the address of the beginning of the program just loaded. DOS calls the program (via a JSR) at that address.

The second monitored location is the INIT vector, at 738–739 (hex $2E2–$2E3). This vector works much the same as the RUN vector, but DOS initializes and checks it for *each segment* as the segments are loaded. If the INIT vector's contents are altered, then DOS assumes the user program wants to stop the load process long enough to call a subroutine. So DOS calls (via a JSR) at the requested address, expecting that the subroutine will return so that the rest of the load can take place. This is a *very* handy feature. Most of you have probably seen it at work—for example, when a program first puts up an introductory screen (maybe just a title and a *Please wait* message) when you run (or boot), then continues to load.

Taking Full Control

The other important difference between the RUN and INIT vectors is that DOS leaves channel 1 open while the INIT routine is called. (DOS always opens and loads the binary file via this channel.) I suppose a really tricky program could close channel 1, open a different binary file, and then return to DOS. DOS would proceed to load the new file as if it were continuing the load of the original one. Most of the time, though, INIT routines should not touch channel 1.

As noted, when you SAVE a binary file from DOS 2.x (and many of its variants), you are allowed to specify both an INIT and a RUN address. But the INIT address is sort of useless, since it is added to the end of a file; so, for example, your opening screen display won't occur until the entire file is loaded. To take full control, you must resort to assembly language (or to a compiled language, such as *Kyan Pascal* or OSS's *Action*). For those of you familiar with assembly language, I present the skeleton listing below. This listing is compatible with the Atari *Assembler Editor* cartridge or the *MAC/65* assembler. You will need to make a handful of minor substitutions if you are using some other assembler.

I'm not going to explain the program in great detail—the source code is fairly well documented. A couple of important points though: Notice that there is no special command to the assembler that will force it to put in an INIT vector (or RUN vector—unless you have the *AMAC* assembler). Instead, we simply create a binary file segment that is only two bytes (one word) long. And this segment is loaded by DOS's loader at—where else—the appropriate vector. So the very act of loading the specified addresses modifies the contents of the vector. What could be neater?

As mentioned, this is *strictly* a program skeleton. It will do nothing as is. You must add some of your own assembly language to it to make it actually do something. So, if you thought INIT and RUN vectors were beyond you, try this skeleton and be ready to change your mind.

INIT Vector Example

```
; Skeleton of a program which puts
; a 'please wait' type message on
```

```
; the screen before loading the
; main code.
;
    *= $3000 ; or someplace
DOINIT
;
; the code which follows is for
; demo purposes only! Use your
; own code...pretty display lists
; or dazzling colors or whatever
;
    LDX   #0          ; channel zero
    LDA   #9          ; Put Text com-
                        mand
    STA   $342        ; command byte
    LDA   #MSG&255
    STA   $344        ; low byte, addr
                        of msg
    LDA   #MSG/256
    STA   $345        ; high byte, ditto
    LDA   #255        ; use a too-big
                        length...
    STA   $348        ; since RETURN
                        terminates
                      ; this call
                        anyway
    JSR   $E456       ; call CIO
    RTS               ; back to DOS
MSG .BYTE 125 ; (clear screen)
    .BYTE 29,29,29,29 ; (cursor down)
    .BYTE 127 ; (tab once)
    .BYTE "—please wait—"
    .BYTE 155 ; (return...end of msg)
;
; now the INIT VECTOR forces DOS
; to call our DOINIT routine
;
    *= $2E2 ; init vector
    .WORD DOINIT ; gets pointed to us
;
; Your main program...
; you are on your own here!
;
    *= $3000 ; the same address if you
               like
;
; I can use the same address because
; my init code can disappear when
; its job is done. This may not
; work with your code. Be careful.
;
DORUN
    ...
    ...
;
; then we get DOS to run our program
; by using a RUN vector.
;
    *= $2E0           ; AMAC uses ORG,
                        not *=
    .WORD DORUN       ; AMAC uses
                        WORD, no dot
    .END              ; AMAC uses END,
                        no dot
```

©

IBM Personal Computing

Donald B. Trivette

The Mother Load Of Software

Imagine thousands and thousands of computer programs available for less than ten cents each, and you have just imagined PC-Sig's new CD-ROM disc with more than 15,000 files. PC-Sig is an unofficial keeper (there being no *official* keeper) of the DOS computer programs that have found their way into the public domain either from computer clubs and savvy individuals or from professional software developers seeking to avoid the high cost of promotion and advertisement.

For about $20 you can join PC-Sig and receive a directory listing the contents of more than 700 disks as well as a monthly newsletter describing new contributions. You may order disks through the mail, by toll-free telephone, or from several dozen PC-Sig authorized dealers. Each disk—even the ones stuffed with 20 or more programs—is priced at just $6. Until recently PC-Sig's only method of distribution was via floppies, but now the distribution has entered the CD-ROM age.

A Huge "Hard Disk"

For $195 you can purchase the entire PC-Sig collection containing thousands of programs all on one CD-ROM disc. The disc comes with driver software causing your CD-ROM player to emulate a huge hard disk, which permits many standard DOS commands to be used to access and manipulate files on the CD-ROM. The CD is organized so that each floppy is allocated its own directory. To read a file named CASTLE.DOC on what would be disk number 47, for example, you simply use the DOS TYPE command. The syntax to point to the subdirectories and display the file would be TYPE D:\1-100\DISK-047\CASTLE.DOC. The DOS COPY command is used to copy the programs from the CD-ROM to your floppies or hard disk for execution, although some programs will execute directly off the disc.

In order to make the CD available as economically as possible, PC-Sig has not included expensive search software with the disc. Instead you must rely on the printed directory, on the DOS FIND command, or on your own word processing software to scan the index files and locate programs that are of interest to you. This isn't quite like looking for a disk in a haystack, since many of the disks have a theme: games, utilities, languages, word processing, communications, and special interest.

The quality of the software runs from ho-hum to excellent. The following descriptions are quoted from the PC-Sig newsletter's hit parade of disks. Disk 517: "IMAGEPRINT allows the production of high-quality characters on a dot-matrix printer....All the mathematical symbols, international characters, and graphics characters are included." Disk 418: "HARD DISK UTILITIES is a collection for the hard-disk user compiled from over 25 disks in our library." Disk 523: "SIDEWINDER is a program that allows printers to output sideways....It works much like the commercial program...is written in PASCAL and the source code is provided." Disk 558: "PC-PROMPT is a memory-resident DOS extension that provides syntax prompting for DOS commands as you type." Disk 273: "BEST UTILITIES have been taken from other library volumes...to collect on one disk all of the better utilities." Disk 310: "QMODEM is a fantastic telecommunications program...." Disk 376: "PATCHES are programs that allow you to place the indicated programs on your hard disk or to make backup copies."

Personal Bests

Some of my own favorites: Disk 53, which contains BASIC programs to make different sounds, including chirp, bomb, siren, engine, and ta-daa; Disk 78, the *PC-Write* word processing program; Disk 120, a PC Chess program; Disk 216, a group of C utilities; Disk 241, specializing in games for the PCjr; Disk 321, home applications; Disk 354, another disk of games just for PCjrs; Disk 372, a collection of dozens of BASIC subroutines; Disk 375, a group of Pascal utilities; and Disks 528–529, which contain the New York Word word processing program. Other disks that look interesting include Disk 447—THE SKY, Disk 459—AGRICULTURAL PROGRAMS, Disk 465—FAMILY TIES, Disks 494–496—THE WORLD DIGITIZED, and Disk 565—HAMRADIO.

Although the PC-Sig CD is quite a bargain, most software distribution will continue to be made on floppy disks until CD-ROM players fall in price. For more information, write PC-Sig, 1030 East Duane Ave., Suite D, Sunnyvale, CA 94086, or call (408) 730-9291.

Fix It Yourself

It wasn't long after I got my IBM PC that I took some of the key caps off just to see what made the keys click. The A and the S caps reseated perfectly, but the space bar didn't quite snap into place, and I'd been working with it partly attached for years. Now—thanks to a new book, *How to Repair and Maintain Your IBM PC*, by Gene Williams—I've been able to repair my faulty keyboard. If you are do-it-yourself inclined (or stupidly curious, as I was), this book may be just what you're looking for. It has chapters on diagnosing what is wrong, disk drives, power supplies, troubleshooting memory, adding to your system, and—when all else fails—dealing with the technician. It's priced at $13.50 from Chilton Publishing in Radnor, PA. ©

Tower Of Babel

This month we'll take a whirlwind tour of some popular ST languages, translating a short but useful program into each language in turn. The assembly language version of this program is only 59 bytes long, but it can speed up disk save operations by a factor of about 30–50 percent, depending on the size of the file involved. No, it's not done with mirrors. In fact, the job is so easy as to be almost trivial from a programming standpoint.

Like some other computers, the ST automatically verifies the success of every disk write operation. At memory location $444 (1092) is a word-length variable that indicates whether verification is in effect. If this flag contains a nonzero value ($FF00 is normal) the ST verifies all disk saves; if it contains zero, verification is turned off. Thus, you can disable verification simply by putting a zero into $444. Our programmers use this technique regularly to speed up saves on floppy disk drives; however, I advise against using it with any hard disk drive.

Assembly Language Version

Program 1 is the source code for the original version, which is written in assembly language. If you don't have an assembler, you can create this program with Program 6. Type in that program with *ST BASIC* and run it; then go to the desktop and double-click on QUICKSAVE.PRG. Verification is disabled, and you should notice a significant speed-up in disk saves.

The first four instructions in Program 1 call the XBIOS routine known as Supexec, which executes a routine in supervisor mode. (As explained in a previous column, certain ST memory areas can be accessed only in supervisor mode.) The first instruction passes to Supexec the address of the routine we want to execute. The second instruction passes the opcode (38) of the Supexec routine itself. When we invoke the routine with trap #14, the machine flips into supervisor mode, performs the designated routine (mycode), and switches back to user mode. In the mycode routine, the instruction clr.w $444 clears, or stores a zero in, location $444. After returning from the XBIOS trap, we add six bytes to the stack pointer to adjust for the word and longword previously pushed onto it. Finally, the instructions clr.w -(sp):trap #1 call Term, the standard GEMDOS routine for terminating a program.

C Version

After writing and testing the assembly language version, I translated it into C (Program 2). The #include statement in the first line tells the compiler to include, or read, a header file named osbind.h when it compiles this program. This particular header file contains definitions for all of the XBIOS, BIOS, and GEMDOS functions on the ST, including Supexec, the XBIOS function we need. Actually, we need to grab only two statements from osbind.h:

```
extern long xbios( );
#define Supexec(a) xbios(38, a)
```

The #define statement allows us to substitute the descriptive name Supexec for XBIOS function 38. We could have skipped the #include and defined Supexec with these statements, but using the header file saves typing and minimizes the risk of typos—important considerations in longer programs, which may use dozens of different system routines. (By the way, every language package contains all the requisite include files.)

Note how the use of a descriptive name makes this program easier to read than the first example. If you know what Supexec does (ignoring for the moment the question of how one attains that knowledge), you can tell at a glance what's involved in any statement where that name appears.

The second statement in the program—extern int mycode( )—makes it possible for Supexec to find the address of mycode, the function we wish to execute in supervisor mode.

The third line in Program 2 declares a pointer variable named ptr. Because C has no keyword equivalent to BASIC's POKE, we must use a pointer, which is simply a variable that points to something else. The first statement in the mycode function is ptr = (int *) 0x0444. It makes ptr point at location $444, or 0x0444 in C terminology. The expression *(int *)* is a cast which tells the compiler we're dealing with a word-length object rather than something of another size. Once ptr is aimed at the right spot, the statement *ptr = 0 stores a zero in the place where it points.

The main body of every C program is contained in a function named main. The curly braces { and } mark the extent of main, and of every other function. Our main function contains the single statement Supexec(mycode), which invokes the Supexec function, passing to it the address of the routine we wish to execute in supervisor mode. The program terminates when we hit main's second curly brace. Notice that we don't have to do anything special to terminate the program; the compiler handles that detail for us, as it does many others.

Pascal Version

Program 3 is the same program written for *Personal Pascal*, the OSS implementation of Pascal for the ST. Pascal is very different from C. In the first place, Pascal originated as an academic, not a practical,

computer language, and it was developed for large, multiuser computers where tinkering with the machine's innards is a definite no-no. Accordingly, the pure incarnation of Pascal forbids any direct access to the computer's memory. But such concerns are less important on a single-user, non-multitasking computer like the ST. And, as a practical matter, most Pascal *compilers* let you do a number of things that the Pascal *language* doesn't want you to do. So let's be naughty.

Near the top of Program 3, the VAR statement declares the variable *ssp*, which we'll use later to store an address. Compare this to the statement which declares ptr in the C program. Though the syntax is slightly different, the result is the same: Both declarations tell the compiler the name and type of a variable which we intend to use. Unlike BASIC, which automatically creates variables as soon as you use them, Pascal and C require you to declare every variable (state its name and type) before use.

The FUNCTION declaration gives the compiler the information it needs to call a system routine—in this case, the GEMDOS function named Super, whose opcode is $20. Again, despite some syntactical variations, you can see the similarity between this and the #define statement which we could have used in the C version.

The naughty part of Program 3 is found in the procedure wpoke, which performs the equivalent of a POKE by means of an unusual variant record named funny. I can't take credit for this procedure, by the way; it comes from an unsupported OSS include file (*unsupported* meaning that OSS offers this code for general use, but does not answer questions or offer other customer support relating to it).

The main body of this program occurs in the last BEGIN-END construct. Just as curly braces enclose the body of a C function, the words BEGIN and END enclose the body of a Pascal procedure. The first statement in this procedure invokes the system routine Super, passing it a zero to get us into supervisor mode and saving the previous address of the user stack pointer in the variable ssp. The second statement calls the procedure wpoke to store a zero in location $444. The third statement calls Super a second time, passing it the address stored in ssp to put us back in user mode. The two calls to Super have the same effect as one call to Supexec, without the difficulty of passing the address of one Pascal procedure to another.

ST BASIC Version

Program 4, the ST BASIC version, requires only one line of code. The DEFDBL statement insures that we'll be POKEing a word-length quantity rather than a byte. Notice that we needn't do anything to put the computer in supervisor mode before doing the POKE: Either ST BASIC itself runs in supervisor mode, or it shifts in and out of supervisor mode to do the POKE.

That may sound convenient, particularly since ST BASIC offers no means to access the XBIOS or GEMDOS routines that invoke supervisor mode. But it makes POKE a potent weapon, indeed. One of the most common and most deadly BASIC programming errors comes from POKEing to an address different from the one intended. In this program, for instance, say that you accidentally type POKE AA, 0 instead of POKE A, 0. The variable AA is never defined in this program, so it has the value zero by default. The effect is to POKE a zero into location zero: ST BASIC crashes with two cherry bombs on the screen, and the system locks up completely when the desktop reappears. Be *extremely* careful with POKE in ST BASIC.

GFA BASIC Version

GFA BASIC offers two different types of POKE statements. POKE, DPOKE, and LPOKE let you store a byte, word, or longword value, respectively, in any memory location that's accessible in user (normal) mode. If you need to access protected memory, you may use SPOKE, SDPOKE, or SLPOKE, to store a byte, word, or longword in a protected address. The *S* in these commands stands for supervisor mode.

What's nice about this scheme is that it protects the unwary tyro against simple blunders, without denying sophisticated programmers access to the machine. If you accidentally POKE to a protected memory address, *GFA BASIC* traps the error and puts up a message suggesting that you check your POKEs and PEEKs. BASIC recovers without crashing, as it should from any run-time error. If you go to the trouble of putting an *S* in front of the POKE, it is assumed that you know what you're doing and are prepared for the possible consequences.

Bloody But Unbowed

That concludes our pocket tour of Babel, but the list of ST dialects is by no means exhausted. Had space (and my patience) permitted, we might have tried Modula-2, Forth, BCPL, and others. It's interesting to see how various languages favor different solutions to the same problem, but don't worry if some of the examples look confusing. Few programmers need to become proficient in more than one or two languages, and the plain truth is that a good programmer can write effective programs in almost any language. So find one that suits your own needs and go to work.

Program 1: Assembly Language Version

```
move.l #mycode, -(sp)
move.w #38, -(sp)
trap #14
addq.l #6,sp

clr.w -(sp)
trap #1

mycode:
clr.w $444
rts
```

Program 2: C Version

```
#include <osbind.h>
extern int mycode( );
int *ptr;

main( )
{
  Supexec(mycode);
}

mycode( )
{
  ptr = (int *) 0x0444;
  *ptr = 0;
}
```

Program 3: Pascal Version

```
PROGRAM quiksave;

VAR
  ssp: long-integer;

FUNCTION super(sp: long-integer):
long-integer;
```

```
  GEMDOS($20);
{$P-}
PROCEDURE wpoke(address: long-
integer; value: integer);
  TYPE
    int-ptr = ^integer;
  VAR
    funny: RECORD
      CASE boolean OF
        true: (a: long-integer);
        false: (p: int-ptr);
      END;
  BEGIN
    funny.a := address;
    funny.p^ := value;
  END;
{$P=}
BEGIN
  ssp := super(0);
  wpoke($444,0);
  ssp := super(ssp);
END
```

Program 4: ST BASIC Version

```
10 defdbl a:a=&H444:poke a,0
```

Program 5: *GFA BASIC* Version

```
sdpoke &H444, 0
```

Program 6: QUIKSAVE.PRG Filemaker

```
100   close:open "R",1,"A:\QUIK
      SAVE.PRG",59
110   field #1,59 as a$
120   for j=1 to 59:read byt$
130   byt=val("&H"+byt$)
140   c=c+1:chk=chk+c+byt
150   x$=x$+chr$(byt):next
160   lset a$=x$:put 1,0:close
170   if chk<>3363 then ? "Typi
      ng error":kill "A:\QUIKSA
      VE.PRG"
180   data 60,1A,00,00,00,1A
190   data 00,00,00,00,00,00
200   data 00,00,00,00,00,00
210   data 00,00,00,00,00,00
220   data 00,00,00,00,2F,3C
230   data 00,00,00,12,3F,3C
240   data 00,26,4E,4E,5C,8F
250   data 42,67,4E,41,42,79
260   data 00,00,04,44,4E,75
270   data 00,00,00,02,00
```

©

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."

Apple Magazine Indexer

The Apple version of this filing utility from the April issue (p. 106) is missing its first three lines. To create a working version, add the following to the published listing:

```
#6 0 GOTO 50
DE 1 REM POSITION COMMAND
6C 2 PRINT D$"READ"Z$",R"FP",B"B
     Y: RETURN
```

Euchre

In the Apple version of this game from the March issue, the first four lines are missing from the BASIC listing (Program 3, p. 54). For a complete program, add the following lines:

```
1 PRINT CHR$(4);"BLOAD EUCHRE.B
  IN,A36008"
2 IF PEEK(190*256)=76 THEN PRIN
  T CHR$(4);"PR#A36008":GOTO 4
3 POKE 54,168:POKE 55,140:CALL
  1002
4 POKE 6,0:POKE 7,141:POKE 230,
  64
```

Many owners of IBM PC and compatible computers have had difficulty deciphering the graphics characters used in their version of the game (Program 5, p. 58). To simplify entry, change or add the following lines, which build the graphics from character codes in DATA statements:

```
PL 1005 COLOR 0,6:LOCATE 1,28,0:
        PRINT CHR$(201)STRING$(1
        0,205)CHR$(187)
PI 1010 LOCATE 2,28:PRINT CHR$(1
        86)"  EUCHRE  "CHR$(186)
JC 1015 LOCATE 3,28:PRINT CHR$(2
        00)STRING$(10,205)CHR$(1
        88)
EC 1060 FOR I=0 TO 5:LOCATE 19+I
        ,33:PRINT CHR$(222);:NEX
        T
CB 1070 FOR I=0 TO 5:LOCATE 5+I,
        33:PRINT CHR$(222);:NEXT
QL 1112 RESTORE 1113:FOR I=0 TO
        2:N$(I)="":FOR J=0 TO 19
        :READ A:N$(I)=N$(I)+CHR$
        (A):NEXT J,I
GA 1113 DATA 32,32,219,32,220,32
        ,220,32,222,32,32,220,32
        ,220,220,32,254,32,254,3
        2,222,32,219,32,32,220,2
        20,32,32,32,220
LL 1114 DATA 32,254,32,219,32,25
        4,32,220,32,32,32,219,32
        ,32,32,32,32,219,32,32,3
        2,32,32,219,32,32,32,32,
        32
CI 2218 COLOR 10,2:Y=F*5+4:LOCAT
        E 21,Y:PRINT CHR$(201)CH
        R$(187):LOCATE 22,Y:PRIN
        T STRING$(2,186):LOCATE
        23,Y:PRINT CHR$(200)CHR$
        (188)
II 3070 PRINT CHR$(205)CHR$(187)
        CD$CL$CHR$(186)NL$CD$CHR
        $(186)CD$CL$CHR$(200)CHR
        $(205)
```

The article states that the Atari version (Program 2) will work on an Atari 400. This is true only if the 400 has memory expansion.

Atari Wari

There is an error in the Atari version of this game from the February issue (Program 6, p. 70), and in the WARI.FEB program on the COMPUTE! Disk for January–March. Line 840 should end with THEN 970 rather than THEN 950. As listed, the program will crash with an ERROR 16 (RETURN without GOSUB) after the maximum number of moves in a limited game. No problem occurs in an unlimited game. Thanks to Frank Walters for pointing out this correction.

SpeedView

This 80-column preview enhancement for *SpeedScript* (November 1986, p. 76) should not be confused with the "SPEEDVIEW" *SpeedScript* preview enhancement released earlier by Upstart Publishing, P.O. Box 22022, Greensboro, NC 27420. The latter program is a part of Upstart Publishing's *SPEEDMATE* customizer program for *SpeedScript*.

©

Synthesis

Dan Monaghan

Hang on to your hats, music enthusiasts—this program turns the Commodore 64 into an impressive music synthesizer with full control over the 64's multifaceted sound chip. Beginners and experts alike can have fun playing music and trying out different sounds with this program. And if you're already familiar with the 64's sound capabilities, you'll find "Synthesis" a powerful tool for experimenting with electronic music sounds. The program works with either disk or tape and requires no extra equipment.

When you got your Commodore 64 or 128, you may have heard that its SID (Sound Interface Device) chip is one of the best sound and music devices in any personal computer. That's true, but programming the SID chip can be a complex business, requiring several POKEs to produce just one sound. "Synthesis" unlocks the full potential of the 64's sound-maker, providing you the equivalent of a sophisticated electronic music synthesizer.

Synthesis turns the 64's keyboard into a musical keyboard, so you can play the synthesizer simply by pressing on the computer's keys. The program also provides a convenient, full-featured editor for designing your own sounds and for experimenting. Once you find a sound that you like, it can be saved to disk or tape for future use or revision. In this way you can build a complete library of instrument voices and sound effects.

Even if you don't know anything about programming the SID chip, you can have fun with this program immediately. This article includes 36 preset *patches* (sound settings), ranging from conventional musical instruments like the flute and cello to far-out electronic sounds such as *space bass, percolator,* and *metallica.*

A Synthesizer And 36 Voices

This article includes two programs. Program 1 is the synthesizer and sound editor. Program 2 is not actually a program, but a data file of 36 different synthesizer voices. While it's not absolutely necessary to type in Program 2, you'll probably want to have it as a demonstration of the wide capabilities of Synthesis and the SID chip.

Both programs must be entered with the "MLX" machine language entry program found elsewhere in this issue. Follow the MLX instructions carefully. If you are using a cassette drive, you'll want to save Program 2 immediately after Program 1 on the same tape. Here are the addresses you need to type the programs in with MLX:

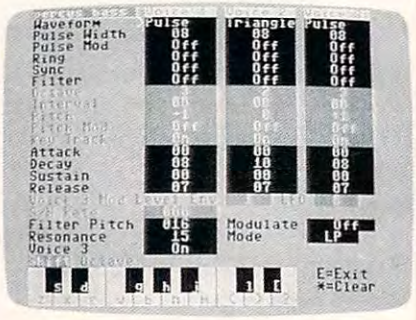
Program 1. SYNTHESIS

Starting address: 0801
Ending address: 1C37

Program 2. VOICES

Starting address: 1C38
Ending address: 23E2

Although it's written in machine language, Synthesis loads and runs the same way a BASIC program does. Load it from disk or tape, type RUN, and press RETURN. Do not try to start it with a SYS command.



"Synthesis" turns the Commodore 64 into an impressive musical synthesizer, and offers full control of the sound chip.

Quick Demo

We'll describe all of the synthesizer's functions fully, but for those who can't wait to try it out, here's a quick demonstration. After typing and saving Programs 1 and 2, load and run Program 1. Synthesis puts you in the file editor screen. The top portion of the screen contains prompts that show you which keys to press for various options. The remainder of the screen is taken up with blank slots which will be filled in after you load a voice file. (Program 2 is a sample voice file.)

Let's begin by playing the synthesizer. From the main screen, press the f7 function key: Synthesis displays the synthesizer screen. At the bottom of the screen is a musical keyboard display that indicates which of the computer's keys act as synthesizer keys. Play the synthesizer using these keys. (If you don't

hear any sound, turn up the volume on your TV or monitor.) This is the default voice—the one that Synthesis uses if you haven't loaded or created a custom voice.

When you've finished playing, press E to exit the synthesizer and return to the file editor. Now let's load a voice file. Press L; then type in the filename when prompted. If you saved Program 2 with the name VOICES, type VOICES and press RETURN. Synthesis then asks whether you wish to load from disk (press D) or tape (press T). After the file has loaded, Synthesis prompts you to press the asterisk (*) key to return to the file editor.

When you return to the main screen, notice that it is now filled with the names of 36 different synthesizer voices. All of these voices have been loaded in memory and are available for your use. To select a voice, press the f1 key. Synthesis displays a cursor (>) in front of the first voice name. Use the cursor keys to move around the screen until you find a voice that sounds interesting. To choose that voice, press RETURN.

The voice which you selected has now been loaded into the synthesizer. To hear what it sounds like, press f7 to go to the synthesizer; then press any of the synthesizer keys. The synthesizer uses the selected voice in place of the default voice which you heard earlier. When you've heard enough, press E to return to the file editor; then press f1 to select one of the other 35 voices. There's a wide variety to choose from.

File Editor

The program begins by displaying the file editor screen. Here is where you select existing voices, name new voices that you have created, and save or load completed files. A voice file can contain as many as 36 individual voices.

The file editor screen offers six options, which you select by pressing the keys indicated on the screen. A list of editor options follows.

f1. The f1 function key loads a voice from the file into the synthesizer. Use the cursor keys to move the pointer to the desired voice, and press RETURN. Synthesis loads that voice into the synthesizer; when you go to the synthesizer screen, that voice is available for your use.

f3. This key takes the voice currently in the synthesizer and stores it in the voice file. If you have just created a new voice and wish to save it, you *must* store the voice with this function before saving the file (the save function is explained below).

f5. The f5 key allows you to change the name of the current voice. Enter a name when prompted and press RETURN. The voice name must be no more than 12 characters in length. To rename an existing voice, first select it with f1; then press f5 to rename it.

f7. Press f7 to exit the file editor and go to the synthesizer screen.

L. Loads a voice file from tape or disk. The program prompts you to enter a filename and then asks whether to load from disk or tape. Press C at the second prompt to cancel the operation.

S. Saves the voice file to tape or disk. This saves all of the voices which appear in the voice list on the file editor screen (voices which have been loaded with L or stored by pressing f3). The current voice in the synthesizer will not be saved unless you have previously stored it in the file.

Synthesizer Functions

The synthesizer screen serves two different purposes: playing music and creating new voices.

By pressing the keys indicated in the musical keyboard display, you can play notes using the current voice parameters. The musical keyboard configuration appears at the bottom of the screen. Press the E key to return to the file editor screen.

The synthesizer screen also allows you to change the current voice characteristics to create a new voice or modify the current one. The voice characteristics appear in the upper portion of the screen.

Playing the synthesizer requires no further explanation. To change a voice characteristic, use the cursor keys to move the pointer to the parameter you wish to change; then press the plus key (+) or minus key (−) to increase or decrease the current value.

When you're using this feature of Synthesis, it will help to have a basic understanding of how the SID chip works—a subject which is beyond the scope of this article. The user's manual for your computer explains more about the SID chip, and many other references are available. If you don't have a complete reference, don't be afraid to experiment: You can't hurt the computer in any way by trying out different settings (although certain combinations may result in no sound or very peculiar sounds). If you produce an unwanted sound, or simply want to discard the current voice, press the asterisk key (*); Synthesis resets all three voices to the default parameters.

Certain features of this synthesizer, such as *sample* and *hold*, are not features of the SID chip itself, but will be familiar to those who have a general knowledge of electronic music synthesis. Following is an explanation of what each voice parameter controls.

Voice Parameters

Waveform. This parameter controls the basic tonal characteristics of each of the synthesizer's three voices. You may choose any of the basic waveforms supplied by the SID chip: triangle, sawtooth, pulse, and noise (random). Note that each of the three voices can have a different waveform.

Pulse width. This parameter controls the symmetry of the pulse waveform. Note that pulse width is relevant only if you have selected a pulse waveform; if you are using some other waveform, changing the pulse width has no effect. The range for this parameter is from 0–8, with 0 creating a very narrow pulse and 8 creating a square wave.

Pulse mod. The pulse mod parameter allows you to use voice 3 to modulate the pulse width (note that pulse width is meaningful only when a pulse waveform is in use). A constantly changing pulse width can create very interesting sounds. You may choose as the source of modulation either the envelope of voice 3 (ENV) or the output of voice 3 (LFO). LFO stands for *low frequency oscillator*, a source which changes with a comparatively low frequency (over a comparatively

long period of time). If you choose ENV as the source of modulation, the modulation is controlled by voice 3's current ADSR parameters (see below).

Ring. This parameter enables or disables ring modulation, a special SID chip effect which combines the frequencies of two voices in a way which produces mathematically incongruous harmonics. If that description sounds baffling, select the *steel drum* voice from the VOICES file and play some notes on the synthesizer. The ringing, metallic tones result from ring modulation. Ring modulation always involves two voices. If you select ring mod for voice 1, then its output is ring modulated with the output of voice 3. Voice 2 is ring modulated by voice 1, and voice 3 is ring modulated by voice 2.

Sync. This parameter enables or disables synchronization, another special effect involving two voices. Synchronization combines the frequency output of two voices to create a more complex sound than would be created with either voice alone. To hear examples of synchronization, select *space bass* or *sync sweep* from the VOICES file. The modulation order for synchronization is the same as for ring modulation. Voice 1 is synchronized with voice 3, voice 2 is synchronized with voice 1, and voice 3 is synchronized with voice 2.

Filtered. This parameter determines whether a voice is routed through the SID chip's built-in filter. The filter allows you to suppress the output of the selected voice within a defined frequency range.

Octave. The SID chip has a frequency range of seven full octaves. This function lets you set any voice to a desired octave. If you set voice 3's octave to 0, that voice goes into LFO (low-frequency oscillator) mode. LFO mode is used in cases where you want to use voice 3's output to modulate some characteristic of a second voice. When you set a voice to LFO mode, that voice produces no audible output itself; instead, its output is rerouted for another purpose.

Interval. The interval parameter causes a voice's frequency to play at a certain number of half-steps above the note being played on the musical keyboard. For instance, if you set voice 1's interval to 7 and play an F# note on the musical keyboard, Synthesis plays a C# note. If you set the voice 1 interval to 0, the voice 2 interval to 4, and the voice 3 interval to 7, you will hear a complete major chord. The *major triad* voice in the VOICES file demonstrates one use of the interval parameter.

Pitch. The pitch parameter allows you to *detune* the selected voice by raising or lowering its pitch slightly, within the range +2 to −2. The idea behind detuning is to make two (or more) voices play the same note, but set one voice just slightly off key by raising or lowering its pitch. The results sound more interesting and "natural" than if both voices were playing in perfect unison. Listen to the *honky tonk* voice in the VOICES file for a demonstration of detuning.

Pitch mod. Pitch modulation is useful for creating vibrato or other pitch-based effects. Again, this parameter always involves two voices. Voice 3 provides a modulating signal which you can use to affect the output of either voice 1 or voice 2. Voice 3 may be in LFO, ENV, or S/H (sample and hold) mode. Sample and hold effects are explained below.

Track. This parameter determines whether or not the designated voice follows the synthesizer keyboard. If tracking is on for a given voice, its frequency is determined by which musical key you press. If tracking is off, the keyboard has no effect on its pitch: Instead, that voice's pitch is controlled solely by its octave, interval, and pitch parameters. Untracking a voice allows you to use its output as an LFO or to create a drone voice which plays at a constant frequency. The *bagpipe* voice in the VOICES file untracks one voice for use as a drone.

Attack. The next four parameters (attack, decay, sustain, and release) are usually abbreviated with the acronym ADSR. Together, they define the *envelope*, or characteristic shape of the final output for a given voice. The attack parameter controls the rate at which, after a musical key is pressed, the level of the designated voice rises to its maximum volume.

Decay. After the attack has reached its peak (see above), the decay parameter controls the rate at which the output of the designated voice drops to the sustain level.

Sustain. This parameter controls the volume level at which the output of the designated voice remains until you release a key on the musical keyboard.

Release. After you've released a key, this parameter controls the rate at which the output of the designated voice fades away into silence.

Mod level. Some of the special effects available in Synthesis involve two voices: One voice is used to modulate (change) the output of a second voice. For pitch mod, pulse mod, sample and hold (S/H), and filter mod, the source of the modulating signal is either the ADSR envelope of voice 3 or the waveform output of voice 3. The mod level parameter controls the intensity of modulation in such cases. If you are using voice 3 to create vibrato, for instance, the mod level can change the vibrato effect from a slight wavering in pitch to a large, multi-octave sweep. The maximum mod level is 9; at this extreme level, you may exceed the range for other parameters, creating a glitch in the sound.

S/H rate. The acronym S/H stands for *sample and hold*, another special electronic sound effect. This feature samples (looks at) the output of voice 3, holds the sampled level, then applies it to the pitch of any voice for which S/H modulation is selected. The modulated voice is then played automatically, just as if you had pressed the key again. Instead of a constantly changing pitch, as with LFO modulation, S/H modulation occurs in discrete steps. If you set the LFO mod level to 0, the pitch of the modulated voice is not changed, but the voice is still automatically rekeyed. For examples of S/H modulation, listen to the *staircase*, *random*, and *mandolin* voices in the VOICES file.

Filter pitch. This parameter sets the resonant frequency of the filter.

Resonance. The resonance parameter controls the strength, or

amount of emphasis, which the filter has.

Mode: This characteristic selects the type of filter to be used. A band-pass filter (BP) causes the filter to pass through, or admit, only frequencies above the designated filter pitch; frequencies above the cutoff point are suppressed. A high-pass filter (HP) passes frequencies above the cutoff point and suppresses lower frequencies. Careful filtering can be very useful in simulating the sounds of natural instruments. However, since the filter is subtractive—that is, it takes away part of the sound you would otherwise hear—it tends to make the final output somewhat quieter than normal.

Voice 3. The voice 3 parameter enables or disables the final output of voice 3. If you are using voice 3 to modulate another voice, you will normally want to disable its output with this feature. If you don't, you may hear unwanted clicks during each envelope cycle for voice 3. If you wish to disable the output of voice 1 or voice 2, set all of the ADSR parameters for that voice to zero.

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

Program 1: Synthesis

```
0801:0B 08 00 00 9E 32 30 36 EC
0809:31 00 00 00 4C E0 0E 00 B3
0811:18 6A E0 00 D0 02 A9 00 C5
0819:A0 00 84 2A 18 90 04 2A E1
0821:26 2A 18 CA D0 F9 66 2A E3
0829:6A A6 14 A4 2A 60 BC 1E 4F
0831:40 B9 CD 14 8D 5F 08 B9 7E
0839:C8 14 8D 60 08 B9 D7 14 55
0841:8D 64 08 B9 D2 14 8D 65 35
0849:08 BD 24 40 F0 08 AC 8D E3
0851:02 B9 EE 17 65 42 7D 18 67
0859:40 7D 1B 40 A8 B9 D4 18 3E
0861:85 45 B9 F0 1A 85 46 60 9F
0869:20 22 0C A5 45 99 00 D4 53
0871:A5 46 99 01 D4 60 BD 0C D8
0879:40 29 FE 20 22 0C 99 04 4E
0881:D4 60 A0 00 B1 26 20 D2 61
0889:FF C8 C0 09 D0 F6 60 A4 3C
0891:28 E6 29 A6 29 18 20 F0 D9
0899:FF A2 02 60 A9 09 18 65 9F
08A1:26 85 26 90 02 E6 27 60 4E
08A9:48 A9 CD 85 26 68 A8 10 8E
08B1:06 A9 E8 85 26 D0 16 F0 36
08B9:14 20 9D 08 C0 01 F0 0D 09
08C1:20 9D 08 C0 02 F0 06 20 56
08C9:9D 08 20 9D 08 4C 83 08 09
08D1:A9 20 20 D2 FF 88 D0 FA AE
08D9:60 8D 51 02 8A 48 A0 02 80
08E1:A9 00 8D 53 02 8D 50 02 96
08E9:A2 11 18 2E 51 02 2E 50 B4
08F1:02 CA F0 11 2E 53 02 AD 55
08F9:53 02 38 E9 0A 30 EB 8D 50
0901:53 02 38 B0 E6 AD 53 02 E5
0909:09 30 99 54 02 88 10 D0 47
0911:68 AA 60 4A 4A 4A 4A 4C 0F
0919:1B 09 20 DA 08 A0 03 20 95
0921:D1 08 C8 20 3D 09 A0 03 8B
0929:20 D1 08 A9 1D 4C D2 FF 1B
0931:20 DA 08 A0 02 20 D1 08 51
0939:A0 00 F0 E7 B9 54 02 20 7B
0941:D2 FF C8 C0 03 D0 F5 60 89
0949:48 A0 02 20 D1 08 68 20 89
0951:D2 FF A0 02 4C D1 08 85 40
0959:26 A9 14 85 27 A2 04 A0 30
0961:00 84 C7 18 20 F0 FF B1 85
0969:26 F0 06 20 D2 FF C8 D0 86
0971:F6 60 A9 06 85 C7 8D 86 99
0979:02 A4 67 A6 68 18 20 F0 E1
0981:FF A9 3E 20 D2 FF 20 E4 83
0989:FF A4 67 A6 68 C9 1D D0 91
0991:0C C0 06 F0 04 A0 06 D0 29
0999:02 A0 1A D0 24 C9 9D F0 99
09A1:F0 C9 11 D0 09 E8 E0 17 92
09A9:D0 17 A2 05 D0 13 C9 91 86
09B1:D0 09 CA E0 04 D0 02 A2 DF
09B9:16 D0 06 C9 0D D0 BA F0 7A
09C1:11 84 67 86 68 A9 9D 20 18
09C9:D2 FF A9 20 20 D2 FF 18 E0
09D1:90 A7 A9 1B 85 8C A5 68 0F
09D9:38 E9 05 AA A9 CE 18 69 EF
09E1:6A 90 02 E6 8C CA 10 F6 A2
09E9:A4 67 C0 06 F0 07 18 69 DD
09F1:35 90 02 E6 8C 85 8B A9 AC
09F9:1C 85 8F A9 38 85 8E 60 5D
0A01:A0 00 20 CF FF 99 00 02 CE
0A09:C8 C9 0D D0 F5 A0 00 B9 8E
0A11:00 02 AA C9 0D F0 1C 29 25
0A19:7F 38 E9 30 B0 04 A9 20 44
0A21:D0 09 8A 29 7F 38 E9 5B CF
0A29:B0 F4 8A 99 00 40 C8 C0 11
0A31:0C D0 DC 4C F8 09 A9 06 25
0A39:8D 86 02 A2 00 A0 00 84 27
0A41:29 18 20 F0 FF BD 00 40 3A
0A49:20 D2 FF E8 E0 0C D0 F5 7F
0A51:A9 0D 85 28 20 90 08 A9 AD
0A59:16 85 27 A9 A9 85 26 85 8E
0A61:C7 A9 00 8D 86 02 A9 10 3C
0A69:85 2A A5 2A 3D 0C 40 D0 8D
0A71:08 20 9D 08 18 06 2A 90 83
0A79:F1 20 83 08 A9 A9 85 26 A4
0A81:CA 10 E3 20 90 08 BD 0F AC
0A89:40 20 1B 09 CA 10 F7 20 60
0A91:90 08 BD 12 40 20 A9 08 A6
0A99:CA 10 F7 20 90 08 BD 0C 44
0AA1:40 29 04 20 A9 08 CA 10 B5
0AA9:F5 20 90 08 BD 0C 40 18 0A
0AB1:29 02 2A 20 A9 08 CA 10 35
0AB9:F3 20 90 08 BD 15 40 20 45
0AC1:A9 08 CA 10 F7 A9 06 8D 07
0AC9:86 02 20 90 08 A0 04 20 99
0AD1:D1 08 BD 18 40 20 0F 0C B6
0AD9:A0 03 20 D1 08 A9 1D 20 61
0AE1:D2 FF CA 10 E8 20 90 08 AA
0AE9:BD 1B 40 20 1B 09 CA 10 50
0AF1:F7 20 90 08 A9 FA 85 26 07
0AF9:A9 16 85 27 BD 1E 40 A8 1B
0B01:F0 18 20 9D 08 C0 01 F0 A9
0B09:11 20 9D 08 C0 02 F0 0A DD
0B11:20 9D 08 C0 03 F0 03 20 AD
0B19:9D 08 20 83 08 CA 10 D4 9C
0B21:20 90 08 A9 16 85 27 BD D9
0B29:21 40 20 A9 08 CA 10 F7 02
0B31:20 90 08 BD 24 40 20 A9 64
0B39:08 CA 10 F7 20 90 08 A9 84
0B41:00 8D 86 02 BD 27 40 18 CE
0B49:20 14 09 CA 10 F6 20 90 6F
0B51:08 BD 27 40 29 0F 20 1B A4
0B59:09 CA 10 F5 20 90 08 BD 19
0B61:2A 40 18 20 14 09 CA 10 0C
0B69:F6 20 90 08 BD 2A 40 29 D5
0B71:0F 20 1B 09 CA 10 F5 A0 2E
0B79:16 20 92 08 A9 06 8D 86 7C
0B81:02 AD 33 40 09 30 20 49 01
0B89:09 A0 20 20 94 08 AD 34 A6
0B91:40 09 30 20 49 09 20 90 51
0B99:08 AD 32 40 20 74 12 A9 0A
0BA1:00 8D 86 02 A0 0D 20 92 18
0BA9:08 AD 2D 40 20 31 09 A0 51
0BB1:1F 20 94 08 AD 2F 40 20 3D
0BB9:A9 08 A0 0D 20 92 08 AD 94
0BC1:30 40 20 1B 09 A0 1F 20 DE
0BC9:94 08 AD 31 40 85 2A E6 48
0BD1:27 A0 27 84 26 A0 00 A9 2E
0BD9:10 25 2A D0 10 A9 06 20 E6
0BE1:9F 08 A9 20 25 2A D0 05 79
0BE9:A9 06 20 9F 08 B1 26 20 C7
0BF1:D2 FF C8 C0 06 D0 F6 A0 98
0BF9:0D 20 92 08 C6 27 A5 2A B9
0C01:29 80 F0 04 A9 00 F0 02 5D
0C09:A9 04 20 A9 08 60 F0 0C 45
0C11:A0 00 18 6A 6A 38 C8 E9 D2
0C19:03 D0 FA 98 09 30 4C D2 44
0C21:FF A0 0E E0 01 D0 02 A0 21
0C29:07 E0 02 D0 02 A0 00 60 3D
0C31:A0 14 84 27 A0 F2 84 26 A1
0C39:A0 00 B1 26 F0 0A 20 D2 FC
0C41:FF C8 D0 F6 E6 27 D0 F2 7D
0C49:20 37 0A A0 00 A2 02 BD D6
0C51:0F 40 99 03 D4 BD 27 40 90
0C59:99 05 D4 BD 2A 40 99 06 81
0C61:D4 98 18 69 07 A8 CA 10 24
0C69:E6 A2 01 BD 2D 40 9D 15 54
0C71:D4 CA 10 F7 18 AD 30 40 40
0C79:0A 0A 0A 0A 85 2A A9 00 23
0C81:AE 15 40 F0 02 09 04 AE 38
0C89:16 40 F0 02 09 02 AE 17 BF
0C91:40 F0 02 09 01 05 2A 8D D4
0C99:17 D4 18 AD 31 40 09 0F FB
0CA1:8D 18 D4 AD 1C D4 AE 33 C0
0CA9:40 20 A4 0D 85 61 84 62 6C
0CB1:86 6D AD 1B D4 AE 34 40 D9
0CB9:20 A4 0D 85 63 84 64 86 81
0CC1:6E CE C1 02 D0 10 AD 32 71
0CC9:40 8D C1 02 A9 01 45 69 03
0CD1:85 69 F0 02 85 6A A2 02 62
0CD9:BD 12 40 F0 16 C9 01 F0 37
0CE1:09 AD 1C D4 AC 33 40 4C B9
0CE9:F1 0C AD 1B D4 AC 34 40 67
0CF1:20 DB 0D CA 10 E2 AD 2F F5
0CF9:40 F0 1C C9 01 D0 09 AC 98
0D01:34 40 AD 1B D4 18 90 06 DA
0D09:AC 33 40 AD 1C D4 18 C0 4E
0D11:09 F0 04 6A C8 D0 F8 18 A6
0D19:6D 2D 40 8D 16 D4 A6 C5 2D
0D21:8A AC 8D 02 F0 02 09 80 9F
0D29:85 F7 BD 00 14 85 6B 10 59
0D31:0A A2 02 20 77 08 CA 10 BC
0D39:FA 30 02 85 41 A2 02 A4 B2
0D41:41 18 84 42 20 2F 08 BD 42
0D49:21 40 F0 22 10 06 A5 6A 92
0D51:F0 24 D0 0F C9 02 D0 0B FA
0D59:A5 62 85 FA A5 61 A4 6D A8
0D61:18 90 08 A5 64 85 FA A5 DB
0D69:63 A4 6E 20 BB 0D 20 69 E9
0D71:08 BD 21 40 10 09 A5 69 80
0D79:D0 05 20 77 08 50 0A A5 F3
0D81:6B 30 06 BD 0C 40 20 7C 18
0D89:08 CA 10 B3 A9 00 85 6A 5A
0D91:A5 6B 10 03 20 3E 0E 4C ED
0D99:A4 0C 18 4C F0 FF 00 00 58
0DA1:00 00 00 18 A8 30 07 85 D6
0DA9:2A 38 A9 7F E5 2A 29 7F BD
0DB1:A0 00 90 01 C8 84 14 4C 0B
0DB9:11 08 18 C0 00 D0 0B 65 2C
0DC1:45 85 45 A5 FA 65 46 85 62
0DC9:46 60 38 85 8D A5 45 E5 F1
0DD1:8D 85 45 A5 46 E5 FA 85 5C
0DD9:46 60 8D 10 02 A9 00 8D 26
0DE1:11 02 88 30 0B 18 6D 10 BC
0DE9:02 90 F7 EE 11 02 B0 F2 FB
0DF1:20 22 0C 99 02 D4 18 AD 01
0DF9:11 02 7D 0F 40 99 03 D4 01
0E01:60 A9 93 20 D2 FF A9 01 17
0E09:8D 86 02 A9 00 85 C7 A0 AE
0E11:0C A2 01 20 9B 0D 86 3D 59
0E19:84 3E A9 3E 20 D2 FF 4C B8
0E21:31 0C A2 00 86 C7 E8 8E E0
0E29:86 02 A6 3D A4 3E 20 9B AB
0E31:0D A9 20 20 D2 FF 18 A5 B0
0E39:B5 F0 01 38 60 A6 F7 8A 28
```

```
0E41:A0 00 29 80 F0 01 C8 84 7C
0E49:B5 8A 29 7F C9 40 F0 EC 1E
0E51:C9 07 F0 30 EA C9 02 D0 88
0E59:03 4C EE 0F C9 28 D0 04 6D
0E61:18 4C 63 10 C9 2B D0 03 A9
0E69:38 B0 F6 C9 31 D0 03 4C 68
0E71:C8 0E C9 0E D0 C6 20 F8 6A
0E79:09 A0 00 84 C6 68 68 A9 DC
0E81:93 4C D2 FF 20 23 0E B0 2F
0E89:09 E0 14 D0 02 A2 00 E8 75
0E91:D0 0C C0 1F D0 01 88 E0 9F
0E99:01 D0 02 A2 15 CA 86 3D F2
0EA1:8A 29 10 F0 08 A0 0C E0 1A
0EA9:10 D0 02 A0 15 84 3E 20 A3
0EB1:9B 0D A9 3E 20 D2 FF CE 13
0EB9:01 24 D0 FB A9 30 8D 01 63
0EC1:24 CE 00 24 D0 F1 60 A0 95
0EC9:34 B9 B8 17 99 00 40 88 CC
0ED1:10 F7 68 68 4C 31 0C 00 C6
0ED9:00 00 00 00 00 00 00 A2 98
0EE1:FF 9A 20 F8 09 A2 00 86 91
0EE9:69 A0 00 B9 B8 17 91 8E 52
0EF1:C8 C0 35 D0 F6 98 18 65 06
0EF9:8E 85 8E 90 02 E6 8F E8 4D
0F01:E0 25 D0 E5 A9 93 20 D2 FF
0F09:FF A9 05 8D 20 D0 8D 21 8B
0F11:D0 A9 0E 20 D2 FF A0 17 B4
0F19:A9 00 99 00 D4 88 10 FA 23
0F21:A9 05 85 68 A9 06 85 67 64
0F29:A9 38 85 8B A9 1C 85 8C E8
0F31:20 F8 09 A0 34 B1 8B 99 E1
0F39:00 40 88 10 F8 A0 01 B9 7F
0F41:39 17 F0 06 20 D2 FF C8 55
0F49:D0 F5 A0 00 B9 00 40 20 CF
0F51:D2 FF C8 C0 0C D0 F5 A9 37
0F59:0D 20 D2 FF A9 20 A2 28 9B
0F61:20 D2 FF CA D0 FA E8 86 BB
0F69:8D 86 C7 A9 06 8D 86 02 F8
0F71:A5 8D 20 1B 09 A0 00 8C D2
0F79:86 02 A9 9D 20 D2 FF B1 68
0F81:8E 20 D2 FF C8 C0 0C D0 7B
0F89:F6 18 A9 35 65 8E 85 8E B0
0F91:90 02 E6 8F E6 8D A5 8D 94
0F99:C9 25 D0 CF A5 CB C9 04 F0
0FA1:D0 0B A9 5C 20 58 09 20 7A
0FA9:73 09 4C 31 0F C9 05 D0 DA
0FB1:15 A9 72 20 58 09 20 73 AF
0FB9:09 A0 34 B9 00 40 91 8B 56
0FC1:88 10 F8 4C 3E 0F C9 06 D3
0FC9:D0 0B A9 90 20 58 09 20 E5
0FD1:01 0A 4C 3E 0F C9 03 D0 D6
0FD9:06 20 02 0E 4C 3E 0F C9 67
0FE1:0D D0 03 4C CB 12 C9 2A 44
0FE9:D0 B2 4C DD 12 20 23 0E E9
0FF1:E0 11 D0 03 4C AE 0E E0 29
0FF9:14 F0 F9 E0 10 D0 0A A0 24
1001:15 C4 3E D0 02 A0 1F D0 53
1009:EB E0 12 D0 0A A0 0C C4 56
1011:3E D0 02 A0 1E D0 DD E0 9F
1019:13 F0 F2 18 A5 B5 F0 01 C5
1021:38 98 B0 0B C0 1E D0 02 6C
1029:A9 02 69 09 A8 D0 C5 C0 31
1031:0C D0 02 A9 27 E9 09 A8 02
1039:D0 BA 10 10 10 10 10 10 64
1041:10 11 11 11 11 11 11 11 E0
1049:11 11 11 11 12 12 88 9F F2
1051:B9 D8 E1 EA F3 0E 28 42 D9
1059:4C 55 69 89 9D BD E7 F3 62
1061:1C 50 A9 00 69 00 85 B5 E4
1069:A6 3D A4 3E 20 73 10 4C DF
1071:D3 0E CA BD 3B 10 48 BD 9C
1079:4F 10 48 A2 00 C0 1E F0 A8
1081:06 E8 C0 15 F0 01 E8 60 06
1089:BD 0C 40 A8 29 0F 85 B5 64
1091:98 29 F0 18 2A D0 02 A9 2A
1099:10 05 B5 9D 0C 40 60 BC 72
10A1:0F 40 A5 B5 D0 09 C8 C0 66
10A9:09 D0 09 A0 00 F0 05 88 04
10B1:10 02 A0 08 98 9D 0F 40 88
10B9:60 BC 12 40 20 C4 10 9D 51
10C1:12 40 60 A5 B5 D0 09 C8 2D
10C9:C0 03 D0 0A A0 FF D0 06 72
10D1:98 10 02 A0 03 88 98 60 58
10D9:BD 0C 40 49 04 9D 0C 40 67
10E1:60 BD 0C 40 49 02 9D 0C C0
10E9:40 60 BD 15 40 49 04 9D 18
10F1:15 40 60 BD 18 40 A4 B5 55
10F9:D0 09 18 C9 54 F0 0B 69 4A
1101:0C D0 07 38 C9 00 F0 02 F3
1109:E9 0C 9D 18 40 60 BC 1B 70
1111:40 A5 B5 D0 09 C8 C0 0C 79
1119:D0 09 A0 00 F0 05 88 10 B6
1121:02 A0 0B 98 9D 1B 40 60 91
1129:BC 1E 40 A5 B5 D0 09 C8 5F
1131:C0 05 D0 09 A0 00 F0 05 8B
1139:88 10 02 A0 04 98 9D 1E C9
1141:40 60 BC 21 40 20 C4 10 61
1149:9D 21 40 60 BD 24 40 49 D8
1151:04 9D 24 40 60 BD 27 40 ED
1159:A4 B5 D0 06 18 69 10 18 54
1161:90 03 38 E9 10 9D 27 40 B7
1169:60 BD 27 40 29 0F A4 B5 98
1171:D0 05 18 69 01 90 03 38 5F
1179:E9 01 29 0F 85 B5 BD 27 8C
1181:40 29 F0 05 B5 9D 27 40 2F
1189:60 BD 2A 40 A4 B5 D0 06 38
1191:18 69 10 18 90 03 38 E9 88
1199:10 9D 2A 40 60 BD 2A 40 03
11A1:29 0F A4 B5 D0 05 18 69 40
11A9:01 90 03 38 E9 01 29 0F 09
11B1:85 B5 BD 2A 40 29 F0 05 EB
11B9:B5 9D 2A 40 60 AD 33 40 C7
11C1:C0 15 F0 03 AD 34 40 A6 3D
11C9:B5 D0 0A AA E8 E0 0A D0 96
11D1:0A A2 00 F0 06 AA CA 10 31
11D9:02 A2 09 C0 15 F0 04 8E D5
11E1:34 40 60 4C A9 13 00 A5 3E
11E9:B5 D0 04 20 5A 12 60 20 99
11F1:63 12 60 C0 0C D0 0C A5 C3
11F9:B5 D0 04 EE 2D 40 60 CE 94
1201:2D 40 60 AE 2F 40 A5 B5 3E
1209:D0 0B E8 E0 03 D0 02 A2 85
1211:00 8E 2F 40 60 CA 10 F9 0B
1219:A2 01 D0 F5 C0 0C D0 13 33
1221:AE 30 40 A5 B5 D0 04 E8 EC
1229:18 90 01 CA 8A 29 0F 8D EE
1231:30 40 60 AD 31 40 A2 00 34
1239:A8 29 80 F0 02 A2 80 86 3D
1241:B5 18 98 29 70 2A 10 02 3A
1249:A9 10 05 B5 8D 31 40 60 54
1251:AD 31 40 49 80 8D 31 40 12
1259:60 EE 32 40 D0 14 A9 05 E2
1261:D0 0D CE 32 40 AD 32 40 8B
1269:38 E9 05 B0 05 A9 FF 8D 2C
1271:32 40 60 38 E9 05 4C 31 7B
1279:09 93 0D 0D 0D 12 D3 41 13
1281:56 45 20 C6 49 4C 45 0D A5
1289:93 0D 0D 0D 12 CC 4F 41 D0
1291:44 20 C6 49 4C 45 0D C5 A4
1299:4E 54 45 52 20 C6 49 4C C2
12A1:45 4E 41 4D 45 20 1F 00 E1
12A9:90 0D 0D D4 41 50 45 2C 4A
12B1:20 C4 49 53 4B 20 4F 52 41
12B9:20 C3 41 4E 43 45 4C 20 D3
12C1:28 D4 2C C4 20 4F 52 C3 A7
12C9:29 00 A0 00 B9 7A 12 20 92
12D1:D2 FF C8 C0 0F D0 F5 A9 D5
12D9:00 48 F0 10 A0 00 B9 89 31
12E1:12 20 D2 FF C8 C0 0F D0 A9
12E9:F5 A9 01 48 A0 00 B9 97 28
12F1:12 F0 06 20 D2 FF C8 D0 17
12F9:F5 A0 00 84 C6 20 CF FF DF
1301:29 7F 99 00 02 C8 C9 0D A2
1309:D0 F3 84 BB A0 00 B9 A9 03
1311:12 F0 06 20 D2 FF C8 D0 38
1319:F5 20 E4 FF C9 54 F0 22 82
1321:C9 44 F0 22 C9 43 D0 F1 6C
1329:68 A0 00 B9 A6 14 F0 06 B4
1331:20 D2 FF C8 D0 F5 A5 CB 1E
1339:C9 31 D0 FA 20 F8 09 4C 9D
1341:3E 0F A2 01 D0 02 A2 08 8A
1349:A0 00 A9 01 20 BA FF A5 96
1351:BB A2 00 A0 02 20 BD FF 14
1359:20 42 14 68 F0 2A A9 00 AC
1361:A2 38 A0 1C 20 D5 FF 90 A5
1369:32 A0 36 C9 04 D0 02 A0 3C
1371:00 C9 05 D0 02 A0 11 C9 36
1379:08 D0 02 A0 17 B9 C2 13 5A
1381:F0 A7 20 D2 FF C8 D0 F5 F5
1389:A9 38 85 C1 A9 1C 85 C2 EA
1391:A9 C1 A2 E2 A0 23 20 D8 2A
1399:FF B0 CE A0 00 B9 48 14 5B
13A1:F0 87 20 D2 FF C8 D0 F5 0E
13A9:E0 00 D0 0A A5 B5 D0 04 A4
13B1:A2 01 D0 02 A2 09 8E 33 2D
13B9:40 60 00 00 00 00 00 00 18
13C1:00 0D 46 49 4C 45 20 4E 8E
13C9:4F 54 20 46 4F 55 4E 44 C5
13D1:0D 00 0D 44 45 56 49 43 BD
13D9:45 20 4E 4F 54 20 50 52 7F
13E1:45 53 45 4E 54 0D 00 00 E3
13E9:0D 4E 4F 20 46 49 4C 45 4B
13F1:20 4E 41 4D 45 0D 00 0D 24
13F9:45 52 52 4F 52 0D 00 FF 5D
1401:FF FF FF FF FF FF FF FF 29
1409:FF FF FF 00 01 FF FF FF 39
1411:FF 03 FF 04 FF FF 02 FF 3E
1419:FF 06 FF 07 08 FF 05 FF 7D
1421:FF 0A FF 0B FF FF 09 FF 8E
1429:FF 0D FF 0E 0F FF 0C FF 06
1431:FF FF FF FF FF FF 10 FF 79
1439:FF FF FF FF FF FF FF FF 61
1441:00 A9 0D 4C D2 FF 00 0D DD
1449:44 4F 4E 45 0D 00 00 00 ED
1451:00 00 00 00 00 00 00 00 79
1459:00 00 00 D3 45 4C 45 43 E7
1461:54 20 D6 4F 49 43 45 20 8D
1469:26 20 D2 45 54 55 52 4E 46
1471:00 D3 45 4C 45 43 54 20 FB
1479:C6 49 4C 45 20 CC 4F 43 4B
1481:41 54 49 4F 4E 20 26 20 DC
1489:D2 45 54 55 52 4E 00 C5 DD
1491:4E 54 45 52 20 CE 41 4D CF
1499:45 20 26 20 D2 45 54 55 DC
14A1:52 4E 20 1F 00 0D 0D D0 9B
14A9:52 45 53 53 20 2A 20 54 2A
14B1:4F 20 52 45 54 55 52 4E 13
14B9:20 54 4F 20 C5 44 49 54 19
14C1:4F 52 00 00 00 00 00 17 3D
14C9:18 18 19 19 FC 68 D4 40 2C
14D1:AC 1A 1A 1A 1B 1B 18 84 B5
14D9:F0 5C C8 00 00 00 88 82 3E
14E1:9C 40 9A 9E 9C 82 8E 90 42
14E9:82 9C 40 00 62 72 70 6C AC
14F1:00 13 05 1D 1D 1D 1D 1D 06
14F9:1D 1D 1D 1D 1D 1D 1D 1D 22
1501:12 D6 4F 49 43 45 20 31 09
1509:20 1D D6 4F 49 43 45 20 5C
1511:32 20 1D D6 4F 49 43 45 D8
1519:20 33 20 0D 90 D7 41 56 B1
1521:45 46 4F 52 4D 0D D0 55 24
1529:4C 53 45 20 D7 49 44 54 B9
1531:48 0D D0 55 4C 53 45 20 8C
1539:CD 4F 44 0D D2 49 4E 47 17
1541:0D D3 59 4E 43 0D C6 49 1C
1549:4C 54 45 52 0D 1F CF 43 44
1551:54 41 56 45 0D C9 4E 54 95
1559:45 52 56 41 4C 0D D0 49 1B
1561:54 43 48 0D D0 49 54 43 F7
1569:48 20 CD 4F 44 0D CB 45 A1
1571:59 20 D4 52 41 43 4B 0D CA
1579:90 C1 54 54 41 43 4B 0D E6
1581:C4 45 43 41 59 0D D3 55 D7
1589:53 54 41 49 4E 0D D2 45 C0
1591:4C 45 41 53 45 0D 1F D6 04
1599:4F 49 43 45 20 33 20 CD 56
15A1:4F 44 20 CC 45 56 45 4C AF
15A9:20 C5 4E 56 1D 1D 1D 1D 39
15B1:1D 1D 20 CC C6 CF 0D D3 E5
15B9:2F C8 20 D2 41 54 45 0D D1
15C1:90 C6 49 4C 54 45 52 20 50
15C9:D0 49 54 43 48 1D 1D 1D 7B
15D1:1D 1D 1D 1D 1D 1D 1D CD AC
15D9:4F 44 55 4C 41 54 45 0D 1F
15E1:D2 45 53 4F 4E 41 4E 43 7D
15E9:45 1D 1D 1D 1D 1D 1D 1D 28
```

```
15F1:1D 1D 1D 1D 1D 1D CD 4F AF
15F9:44 45 ØD D6 4F 49 43 45 12
1601:20 33 ØD 1F 12 D3 48 49 57
1609:46 54 92 20 CF 43 54 41 37
1611:56 45 ØD 12 Ø5 20 20 90 F5
1619:20 20 Ø5 20 90 20 20 Ø5 4A
1621:20 A7 20 20 90 20 20 Ø5 97
1629:20 90 20 20 Ø5 20 90 20 79
1631:20 Ø5 20 A7 20 20 90 20 EF
1639:20 Ø5 20 90 20 20 Ø5 20 6F
1641:A7 90 92 20 20 C5 3D C5 12
1649:58 49 54 ØD 12 Ø5 20 20 54
1651:90 20 53 Ø5 20 90 20 44 50
1659:Ø5 20 A7 20 20 90 20 47 D1
1661:Ø5 20 90 20 48 Ø5 20 90 53
1669:20 4A Ø5 20 A7 20 20 90 69
1671:20 4C Ø5 20 90 20 5B Ø5 24
1679:20 A7 90 92 20 20 2A 3D ED
1681:C3 4C 45 41 52 ØD Ø5 12 42
1689:20 5A A7 20 58 A7 20 43 38
1691:A7 20 56 A7 20 42 A7 20 58
1699:4E A7 20 4D A7 20 3C A7 8D
16A1:20 3E A7 20 3F A7 ØØ ØØ FC
16A9:D4 52 49 41 4E 47 4C 45 7F
16B1:1D D3 41 57 54 4F 4F 54 D1
16B9:48 1D DØ 55 4C 53 45 20 1B
16C1:20 20 1D CE 4F 49 53 45 22
16C9:20 20 20 1D 20 20 20 CF 75
16D1:46 46 20 20 1D 20 20 20 82
16D9:CC C6 CF 20 20 1D 20 20 EF
16E1:20 C5 4E 56 20 20 1D 20 9A
16E9:20 20 D3 2F C8 20 20 1D BF
16F1:20 20 20 CF 4E 20 20 20 8A
16F9:1D 20 20 20 2D 32 20 20 55
1701:20 1D 20 20 20 2D 31 20 C4
1709:20 20 1D 20 20 20 20 30 E6
1711:20 20 20 1D 20 20 20 2B 1A
1719:31 20 20 20 1D 20 20 20 B7
1721:2B 32 20 20 20 1D 20 20 4D
1729:CC DØ 20 20 20 20 C2 DØ CF
1731:20 20 20 20 C8 DØ 20 20 67
1739:93 13 12 1F 20 46 31 20 C6
1741:92 20 CC 4F 41 44 20 D6 81
1749:4F 49 43 45 20 20 12 20 F3
1751:46 33 20 92 20 D3 54 4F E4
1759:52 45 20 D6 4F 49 43 45 DE
1761:ØD 12 20 46 35 20 92 20 72
1769:CE 45 57 20 CE 41 4D 45 98
1771:20 20 20 20 12 20 46 37 92
1779:20 92 20 D3 59 4E 54 48 92
1781:ØD 12 20 20 CC 20 92 20 EC
1789:CC 4F 41 44 20 C6 49 4C 59
1791:45 20 20 20 12 20 20 D3 95
1799:20 92 20 D3 41 56 45 20 CB
17A1:C6 49 4C 45 ØD Ø5 C3 55 BC
17A9:52 52 45 4E 54 20 D6 4F 43
17B1:49 43 45 20 12 ØØ ØØ 2A BA
17B9:2A 2A 2A 2A 2A 2A 2A 2A E7
17C1:2A 2A 2A 11 11 11 Ø8 Ø8 CA
17C9:Ø8 ØØ ØØ ØØ ØØ ØØ ØØ 24 20
17D1:30 3C ØØ ØØ ØØ Ø2 Ø2 Ø2 35
17D9:ØØ ØØ ØØ Ø4 Ø4 Ø4 ØØ ØØ 78
17E1:ØØ F8 F8 F8 20 ØØ ØØ ØF ØD
17E9:1Ø ØA Ø1 ØØ ØØ ØØ ØC ØØ DA
17F1:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ 20
17F9:ØØ ØØ ØØ Ø1 Ø6 ØB 11 1B D1
1801:25 34 47 61 83 AE E3 ØA 78
1809:1A 2B 3D 5Ø 63 79 8F A7 85
1811:CØ DB F7 15 35 56 7A AØ 81
1819:C7 F2 1F 4E 81 B6 EE 2A A1
1821:6A AD F4 4Ø 8F E4 3E 9D BE
1829:Ø2 6C DD 55 D4 5A E9 8Ø EA
1831:1F C9 7C 3A Ø4 D9 BB AB 41
1839:A8 B5 D2 ØØ 3F 92 F9 75 33
1841:Ø8 B2 77 56 51 6B A4 ØØ F7
1849:7F 24 F2 EA 1Ø 65 EE AC EF
1851:A3 D6 49 ØØ FF 49 E5 D5 F8
1859:21 CB DC 58 46 AC 93 ØØ 3A
1861:FE 93 CA AB 42 97 B8 Ø2 ED
1869:Ø7 ØC 13 1D 28 37 4B 67 7Ø
1871:8B B8 ED ØB 1B 2C 3E 5Ø 5A
1879:65 7A 9Ø A8 C1 DC F8 16 21
1881:36 58 7C A1 CA F4 21 51 4A
1889:83 B9 F1 2D 6D BØ F8 43 5E
1891:94 E8 42 A2 Ø7 72 E3 5B DD
1899:DA 61 FØ 87 28 D1 85 44 FD
18A1:ØE E4 C7 B7 B5 C3 E1 ØF 16
18A9:5Ø A3 ØB 89 1D C8 8E 6E 7C
18B1:6B 86 C2 1F AØ 47 17 12 E5
18B9:3A 91 1C DD D7 ØD 84 3E Ø7
18C1:41 8F 2F 24 74 23 39 BA FB
18C9:AE 1B Ø8 7D 82 1F 5E 48 86
18D1:E8 47 72 Ø3 Ø8 ØD 15 1F 84
18D9:2B 3B 5Ø 6E 93 C2 FA ØC Ø9
18E1:1C 2D 3E 51 66 7B 91 A9 36
18E9:C3 DD FA 18 38 5A 7D A3 1E
18F1:CC F6 23 53 86 BB F4 3Ø 1D
18F9:7Ø B4 FB 47 98 ED 47 A7 36
1901:ØC 77 E9 61 E1 68 F7 8F 9A
1909:3Ø DA 8F 4E 18 EF D2 C3 CA
1911:C2 D1 EF 1F 6Ø B5 1E 9C BB
1919:31 DF A5 87 85 A2 DF 3E BD
1921:C1 6A 3C 39 63 BE 4B ØE A4
1929:ØB 45 BF 7D 83 D5 79 72 DA
1931:C7 7C 96 1D 17 8A 7E FA E5
1939:Ø6 AB F2 E5 8E F8 2D Ø4 CC
1941:Ø9 ØE 17 21 2E 3F 55 75 FE
1949:9C CD Ø2 ØD 1D 2D 3F 52 BC
1951:67 7C 93 AB C4 DF FB 1A 3B
1959:3A 5B 7F A5 CE F8 26 56 C6
1961:88 BE F7 34 74 B7 FF 4B 97
1969:9C F1 4C AC 11 7D EF 68 81
1971:E8 6F FF 97 38 E3 98 58 48
1979:23 FA DE DØ DØ DF FE 2E 17
1981:71 C6 3Ø BØ 46 F5 BD AØ 55
1989:AØ BE FD 5D E2 8D 61 6Ø C1
1991:8D EA 7A 4Ø 4Ø 7D FA BB 42
1999:C5 1B C3 C1 1A D4 F4 81 99
19A1:8Ø FA F4 77 8A 37 87 83 AC
19A9:35 A9 E9 Ø5 ØA ØF 19 23 5Ø
19B1:31 43 5B 7C A5 D8 ØA ØD 32
19B9:1D 2E 4Ø 53 68 7D 94 AC 52
19C1:C5 EØ FD 1B 3B 5D 81 A7 7A
19C9:DØ FB 28 58 8B C1 FA 37 7E
19D1:77 BB Ø3 4F AØ F6 5Ø B1 37
19D9:16 82 F5 6E EE 76 Ø6 9F 5A
19E1:4Ø EC A1 62 2D Ø5 EA DC F9
19E9:DD ED ØD 3E 81 D8 43 C4 C6
19F1:5B ØB D4 B9 BA DA 1A 7D BD
19F9:Ø3 B1 87 88 B7 16 A9 72 6F
1A01:75 B4 35 FA Ø7 62 ØE 1Ø 61
1A09:6E 2D 52 E4 EA 69 6A F4 1F
1A11:ØF C4 1C 21 DD 5A A5 ØØ 37
1A19:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ 4D
1A21:ØØ ØØ ØØ Ø1 Ø1 Ø1 Ø1 Ø1 74
1A29:Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø2 5E
1A31:Ø2 Ø2 Ø2 Ø2 Ø2 Ø2 Ø3 Ø3 68
1A39:Ø3 Ø3 Ø3 Ø4 Ø4 Ø4 Ø4 Ø5 8D
1A41:Ø5 Ø5 Ø6 Ø6 Ø7 Ø7 Ø7 Ø8 C4
1A49:Ø8 Ø9 Ø9 ØA ØB ØB ØC ØD 2F
1A51:ØE ØE ØF 1Ø 11 12 13 15 FE
1A59:16 17 18 1A 1C 1D 1F 21 B7
1A61:23 25 27 2A 2C 2F 31 34 AC
1A69:38 3B 3E 42 46 4A 4F 54 C2
1A71:58 5E 63 69 7Ø 76 7D 85 4A
1A79:8D 95 9E A8 B1 BC C7 D3 1C
1A81:E0 ED FB ØØ ØØ ØØ ØØ ØØ 21
1A89:ØØ ØØ ØØ ØØ ØØ ØØ ØØ Ø1 BE
1A91:Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 C5
1A99:Ø1 Ø1 Ø1 Ø2 Ø2 Ø2 Ø2 Ø2 EC
1AA1:Ø2 Ø2 Ø3 Ø3 Ø3 Ø3 Ø3 Ø4 16
1AA9:Ø4 Ø4 Ø4 Ø5 Ø5 Ø5 Ø6 Ø6 FF
1AB1:Ø7 Ø7 Ø7 Ø8 Ø8 Ø9 Ø9 ØA ØD
1AB9:ØB ØB ØC ØD ØE ØE ØF 1Ø 5F
1AC1:11 12 13 15 16 17 19 1A 1Ø
1AC9:1C 1D 1F 21 23 25 27 2A 6F
1AD1:2C 2F 32 35 38 3B 3F 42 FØ
1AD9:46 4B 4F 54 59 5E 64 6A AA
1AE1:7Ø 77 7E 85 8D 96 9F A8 Ø3
1AE9:B2 BD C8 D4 EØ EE FC ØØ ØA
1AF1:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ 26
1AF9:ØØ ØØ ØØ Ø1 Ø1 Ø1 Ø1 Ø1 4D
1B01:Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø2 38
1B09:Ø2 Ø2 Ø2 Ø2 Ø2 Ø2 Ø3 Ø3 42
1B11:Ø3 Ø3 Ø3 Ø4 Ø4 Ø4 Ø4 Ø5 67
1B19:Ø5 Ø5 Ø6 Ø6 Ø7 Ø7 Ø7 Ø8 9E
1B21:Ø8 Ø9 Ø9 ØA ØB ØB ØC ØD Ø9
1B29:ØE ØE ØF 1Ø 11 12 13 15 D8
1B31:16 17 19 1A 1C 1D 1F 21 B1
1B39:23 25 27 2A 2C 2F 32 35 89
1B41:38 3B 3F 43 47 4B 4F 54 D8
1B49:59 5E 64 6A 7Ø 77 7E 86 DB
1B51:8E 96 9F A8 B3 BD C8 D4 ED
1B59:E1 EE FD ØØ ØØ ØØ ØØ ØØ FB
1B61:ØØ ØØ ØØ ØØ ØØ ØØ Ø1 Ø1 9A
1B69:Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 9F
1B71:Ø1 Ø1 Ø1 Ø2 Ø2 Ø2 Ø2 Ø2 C6
1B79:Ø2 Ø2 Ø3 Ø3 Ø3 Ø3 Ø3 Ø4 EF
1B81:Ø4 Ø4 Ø4 Ø5 Ø5 Ø5 Ø6 Ø6 D9
1B89:Ø7 Ø7 Ø7 Ø8 Ø8 Ø9 Ø9 ØA E6
1B91:ØB ØB ØC ØD ØE ØE ØF 1Ø 39
1B99:11 12 13 15 16 17 19 1A E9
1BA1:1C 1D 1F 21 23 25 27 2A 49
1BA9:2C 2F 32 35 38 3B 3F 43 CB
1BB1:47 4B 4F 54 59 5F 64 6A Ø9
1BB9:71 77 7E 86 8E 96 9F A9 76
1BC1:B3 BE C9 D5 E2 EF FD ØØ EA
1BC9:ØØ ØØ ØØ ØØ ØØ ØØ ØØ ØØ FF
1BD1:ØØ ØØ Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 47
1BD9:Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø1 Ø2 11
1BE1:Ø2 Ø2 Ø2 Ø2 Ø2 Ø2 Ø3 Ø3 1B
1BE9:Ø3 Ø3 Ø3 Ø4 Ø4 Ø4 Ø5 Ø5 42
1BF1:Ø5 Ø5 Ø6 Ø6 Ø7 Ø7 Ø7 Ø8 77
1BF9:Ø8 Ø9 ØA ØA ØB ØB ØC ØD Ø2
1C01:ØE ØF ØF 1Ø 11 12 14 15 F4
1C09:16 17 19 1A 1C 1E 1F 21 8F
1C11:23 25 28 2A 2D 2F 32 35 8B
1C19:38 3C 3F 43 47 4B 5Ø 54 F4
1C21:5A 5F 65 6B 71 78 7F 86 B4
1C29:8E 97 AØ A9 B4 BE CA D6 4A
1C31:E2 FØ FE ØØ ØØ ØØ ØØ ØØ F6
```

Program 2: Voices

```
1C38:5Ø 49 41 4E 4F 20 20 20 53
1C40:20 20 20 20 11 21 41 Ø8 2E
1C48:Ø4 Ø4 ØØ ØØ ØØ ØØ ØØ ØØ 83
1C50:3Ø 3Ø 3Ø ØØ ØØ ØØ Ø1 Ø2 B6
1C58:Ø2 ØØ ØØ ØØ Ø4 Ø4 Ø4 ØA D3
1C60:Ø8 ØA Ø8 Ø6 Ø6 44 ØØ ØØ C1
1C68:ØØ 1Ø Ø5 Ø1 ØØ 48 4F 4E 63
1C7Ø:4B 59 20 54 4F 4E 4B 20 58
1C78:20 11 41 41 Ø8 Ø6 Ø4 ØØ A1
1C8Ø:ØØ ØØ ØØ ØØ ØØ 24 3Ø 3Ø D9
1C88:ØØ ØØ ØØ Ø2 Ø4 ØØ ØØ ØØ Ø1
1C9Ø:ØØ Ø4 Ø4 Ø4 ØØ ØA ØA F8 BF
1C98:Ø6 Ø6 1Ø ØØ ØØ ØF 9Ø Ø5 B9
1CAØ:Ø1 ØØ 4D 41 4E 44 4F 4C 85
1CA8:49 4E 2Ø 2Ø 2Ø 2Ø 41 11 34
1CBØ:41 Ø5 Ø8 Ø6 ØØ ØØ ØØ ØØ 2C
1CB8:ØØ ØØ 3C ØØ 3C ØØ ØØ ØØ 5A
1CCØ:Ø4 Ø2 Ø1 FF ØØ FF Ø4 ØØ A3
1CC8:Ø4 Ø6 22 Ø5 Ø6 ØØ Ø5 1Ø 63
1CDØ:ØØ ØØ ØF 1Ø Ø8 Ø1 ØØ 43 73
1CD8:4C 41 52 49 4E 45 54 2Ø B6
1CEØ:2Ø 2Ø 2Ø 11 11 41 Ø8 Ø8 EB
1CE8:Ø8 ØØ ØØ ØØ ØØ ØØ ØØ ØØ 25
1CFØ:ØØ 3Ø Ø7 ØØ ØØ Ø2 Ø2 Ø2 24
1CF8:ØØ ØØ Ø1 ØØ ØØ Ø4 ØØ 22 83
1DØØ:3Ø F8 Ø2 F8 Ø9 ØØ ØØ ØF B7
1DØ8:9Ø Ø5 Ø1 Ø1 53 54 45 45 B7
1D1Ø:4C 2Ø 44 52 55 4D 2Ø 2Ø 66
1D18:11 15 11 Ø8 Ø8 Ø8 ØØ ØØ 23
1D2Ø:ØØ ØØ ØØ ØØ 24 3Ø 3Ø ØØ 9C
1D28:ØØ Ø7 Ø2 Ø2 Ø2 ØØ ØØ ØØ 94
1D3Ø:Ø4 Ø4 Ø4 ØØ Ø9 ØØ F8 Ø8 3Ø
1D38:ØØ 1Ø ØØ ØØ ØF 9Ø Ø5 Ø1 3C
1D4Ø:ØØ 43 45 4C 4C 4F 2Ø 2Ø B8
1D48:2Ø 2Ø 2Ø 2Ø 2Ø 41 21 21 ØA
1D5Ø:Ø8 Ø8 Ø8 ØØ ØØ ØØ ØØ ØØ 91
1D58:ØØ 18 24 24 ØØ ØØ ØØ Ø2 61
1D6Ø:Ø3 Ø1 ØØ ØØ ØØ Ø4 Ø4 Ø4 78
1D68:5Ø 68 57 F8 Ø8 98 1Ø ØØ 22
1D7Ø:ØØ ØF 1Ø Ø5 Ø1 ØØ 53 54 C3
1D78:52 49 4E 47 53 2Ø 31 2Ø ØA
1D8Ø:2Ø 2Ø 21 21 21 Ø8 Ø8 Ø8 4A
```

```
1D88:00 00 00 00 00 00 3C 3C 77
1D90:3C 00 00 00 02 03 01 00 07
1D98:00 00 04 04 04 50 50 50 E5
1DA0:F8 F8 F8 10 00 00 0F 10 E3
1DA8:05 01 00 53 54 52 49 4E A7
1DB0:47 53 20 32 20 20 20 21 6D
1DB8:21 21 08 08 08 00 00 00 8D
1DC0:00 00 00 24 30 3C 00 00 AF
1DC8:00 02 03 01 00 00 00 04 F7
1DD0:04 04 50 50 50 F8 F8 F8 6E
1DD8:10 00 00 0F 10 05 01 00 A2
1DE0:54 55 42 41 20 20 20 20 D8
1DE8:20 20 20 20 11 21 21 08 98
1DF0:08 08 00 00 00 00 04 04 3D
1DF8:18 18 18 00 00 00 02 03 4F
1E00:02 00 00 00 04 04 04 06 7B
1E08:05 05 06 08 D8 04 00 02 22
1E10:0F 10 05 06 00 46 4C 55 DF
1E18:54 45 20 20 20 20 20 20 B7
1E20:20 11 11 11 08 08 08 00 54
1E28:00 00 00 00 00 00 3C 3C 19
1E30:07 00 00 02 02 02 00 01 29
1E38:01 00 04 04 00 43 50 F8 5C
1E40:00 F8 10 00 00 0F 90 05 1F
1E48:01 01 50 45 52 43 55 53 41
1E50:20 42 41 53 53 20 41 11 39
1E58:41 08 08 08 00 00 00 00 B8
1E60:00 00 24 18 24 00 00 00 C3
1E68:03 02 01 00 00 00 04 04 D2
1E70:04 08 0A 08 07 07 07 10 E4
1E78:00 00 0F 10 05 01 00 48 0C
1E80:41 52 50 53 49 43 48 4F 68
1E88:52 44 20 41 41 41 03 01 2D
1E90:03 00 00 00 00 00 00 48 96
1E98:3C 30 00 00 00 02 01 02 0B
1EA0:00 00 00 04 04 04 09 08 67
1EA8:09 05 05 05 10 00 00 0F 2B
1EB0:10 05 01 00 41 43 43 4F 43
1EB8:52 44 49 41 4E 20 20 20 BF
1EC0:11 41 41 08 08 08 00 00 DE
1EC8:00 00 00 00 00 3C 3C 00 6E
```

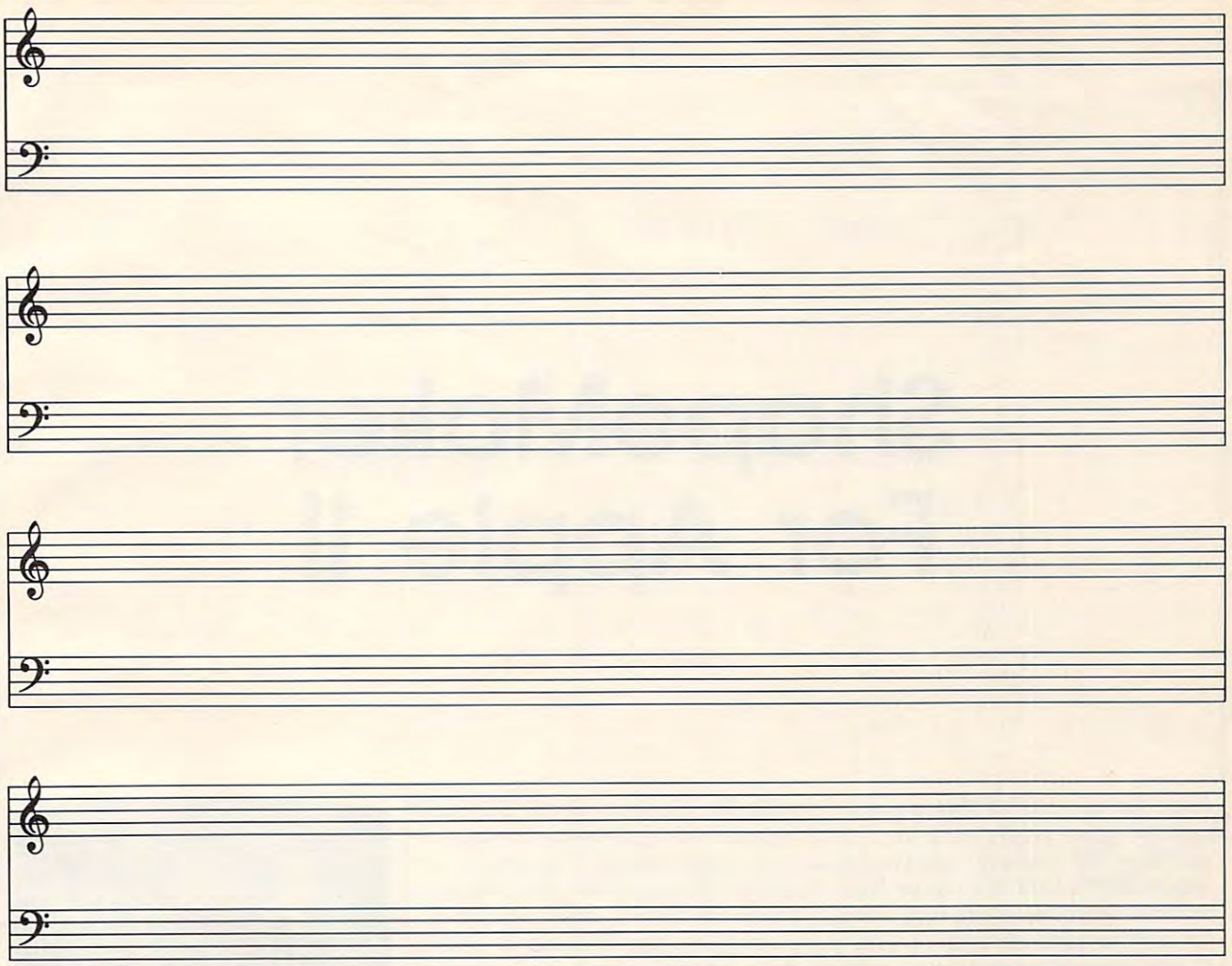
Attention Programmers

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."

```
1ED0:00 00 02 04 00 00 00 00 8D
1ED8:04 04 04 09 20 20 78 F8 94
1EE0:F8 10 00 00 0F 90 05 01 63
1EE8:00 4D 41 52 54 49 41 4E 5E
1EF0:20 46 49 53 48 11 21 21 17
1EF8:08 08 08 00 00 00 00 04 40
1F00:04 00 24 30 09 00 00 02 12
1F08:02 02 02 00 00 00 04 04 14
1F10:C0 00 00 F8 F8 F8 10 00 0A
1F18:01 0F 90 05 03 08 50 45 1B
1F20:52 43 4F 4C 41 54 4F 52 53
1F28:20 20 11 11 11 08 08 08 72
1F30:00 00 00 00 00 00 24 48 FE
1F38:3C 00 00 00 02 02 02 00 B0
1F40:00 00 04 04 04 00 02 04 67
1F48:F8 02 04 20 00 00 0F 90 B4
1F50:05 01 00 46 52 4E 43 48 50
1F58:20 50 4F 4C 49 43 45 41 8C
1F60:11 11 08 08 08 00 00 00 2D
1F68:00 00 00 00 3C 3C 03 07 86
1F70:07 02 04 00 00 01 01 00 39
1F78:04 04 00 00 00 F8 F8 F8 88
1F80:20 00 00 0F 90 05 01 05 5F
1F88:55 46 4F 20 54 41 4B 45 72
1F90:4F 46 46 20 21 15 41 08 BA
1F98:08 08 00 00 00 00 04 04 E8
1FA0:00 30 30 09 0B 0B 02 02 0C
1FA8:02 00 01 02 00 00 00 F0 19
1FB0:00 00 FF FF 00 00 00 02 F0
1FB8:0F 90 0E 08 08 53 59 4E 73
1FC0:43 20 53 57 45 45 50 20 88
1FC8:20 11 13 11 08 01 04 00 1B
1FD0:00 00 00 00 00 00 3C 24 AB
1FD8:06 00 00 04 02 02 02 01 77
1FE0:00 00 04 04 CB 18 00 08 A6
1FE8:87 47 10 00 00 0F 90 05 21
1FF0:02 07 48 45 4C 49 43 4F AC
1FF8:50 54 45 52 20 20 41 81 C7
2000:85 08 08 08 00 00 00 00 86
2008:00 04 24 54 54 00 00 00 B5
2010:02 02 02 FF FF FF 04 04 1E
2018:04 03 04 20 00 00 F0 14 93
2020:00 02 0F 10 0E 09 00 46 9E
2028:49 4C 54 45 52 45 44 20 4F
2030:45 4E 56 11 11 41 08 06 26
2038:08 00 00 00 00 04 04 00 94
2040:24 30 03 00 00 02 02 02 0D
2048:00 00 00 00 04 04 59 03 6E
2050:00 08 5A FA 00 00 02 0F A0
2058:10 05 08 00 48 41 4E 44 0B
2060:20 43 4C 41 50 20 20 20 82
2068:81 81 81 08 08 08 00 00 DA
2070:00 00 00 00 24 30 3C 00 0B
2078:00 00 02 02 02 00 00 00 29
2080:04 04 04 22 53 03 00 03 10
2088:03 10 00 00 0F 10 05 01 12
2090:00 4D 41 4A 4F 52 20 54 49
2098:52 49 41 44 20 41 41 41 8A
20A0:08 08 08 00 00 00 00 00 E7
20A8:00 24 30 30 07 04 00 02 45
20B0:02 02 00 00 00 04 04 04 8E
20B8:00 00 00 F8 F8 F8 10 00 54
20C0:00 0F 10 05 01 00 53 54 1A
20C8:41 49 52 43 41 53 45 20 7C
20D0:20 20 21 11 41 08 08 08 A0
20D8:00 00 00 00 00 00 00 00 19
20E0:3C 01 00 00 02 02 02 00 9B
20E8:00 FF 00 04 04 00 00 09 92
20F0:F8 00 05 10 00 00 0F 90 FD
20F8:57 01 07 52 41 4E 44 4F 46
2100:4D 20 20 20 20 20 20 81 3A
2108:11 41 08 08 08 00 00 00 E4
2110:00 00 00 00 00 30 01 00 15
2118:00 02 02 02 00 00 FF 00 3B
2120:04 04 00 00 09 00 00 05 B2
2128:10 00 00 0F 90 35 01 08 C6
2130:50 41 54 54 45 52 4E 53 1E
2138:20 20 20 20 21 11 41 08 70
2140:08 08 00 00 00 00 00 00 88
2148:00 00 30 01 00 00 02 02 A6
2150:02 00 00 FF 04 04 04 00 CB
2158:00 05 00 00 04 10 00 00 3C
```

```
2160:0F 90 35 01 07 53 50 41 6C
2168:43 45 20 42 41 53 53 20 E3
2170:20 11 43 41 08 05 08 00 E7
2178:00 00 00 00 00 00 24 30 18 C3
2180:00 02 00 02 02 02 00 02 7D
2188:00 04 04 04 07 00 00 07 CB
2190:F8 00 10 00 00 0F 90 05 B3
2198:07 00 42 41 47 50 49 50 19
21A0:45 20 20 20 20 20 41 41 D8
21A8:41 02 02 02 00 00 00 00 6C
21B0:00 00 30 48 48 00 00 00 BF
21B8:02 04 00 00 00 00 00 04 01
21C0:04 00 20 20 F8 F8 F8 10 B8
21C8:00 00 0F 10 05 01 00 54 6E
21D0:52 55 4D 50 45 54 20 20 1C
21D8:20 20 20 11 41 21 08 02 E8
21E0:08 00 00 00 00 04 04 00 3F
21E8:3C 3C 00 00 00 02 02 02 66
21F0:00 00 00 00 04 04 24 25 D0
21F8:55 07 A8 A8 2B 00 02 0F B3
2200:90 0A 06 00 52 4E 44 20 44
2208:48 41 52 4D 4F 4E 49 58 7E
2210:81 43 41 08 08 08 00 00 EE
2218:00 00 00 00 0C 3C 30 00 0E
2220:00 00 02 02 02 00 FF 02 D6
2228:00 04 04 00 00 00 8F FC 0A
2230:00 20 00 00 0F 90 24 06 85
2238:08 42 41 53 53 4F 4F 4E 33
2240:20 20 20 20 20 11 11 41 4B
2248:08 08 08 00 00 00 00 00 93
2250:04 00 00 24 06 00 00 02 0B
2258:02 02 00 00 00 00 00 04 22
2260:00 44 00 F8 02 F8 0D 00 53
2268:00 00 90 05 01 01 4F 52 0C
2270:47 41 4E 20 20 20 20 20 56
2278:20 20 11 11 11 08 08 08 C8
2280:00 00 00 00 00 00 30 3C 61
2288:24 00 07 00 02 02 03 00 DD
2290:00 00 04 04 04 00 08 00 C5
2298:F4 44 D4 20 00 00 0F 10 33
22A0:0A 01 00 57 4F 57 20 20 D7
22A8:20 20 20 20 20 20 20 11 DD
22B0:21 41 08 08 08 00 00 00 97
22B8:00 04 04 24 24 30 00 00 A2
22C0:00 02 02 02 00 00 00 04 E9
22C8:04 04 A9 00 00 08 F8 F8 50
22D0:01 00 02 0F 90 0A 08 00 83
22D8:53 4C 49 44 45 52 20 20 1B
22E0:20 20 20 20 11 11 41 08 9A
22E8:08 08 00 00 00 00 00 00 33
22F0:24 30 30 00 07 00 02 04 99
22F8:01 00 02 02 04 04 04 0A 60
2300:00 00 87 F6 F6 20 00 00 DE
2308:0F 90 0A 07 00 4D 45 54 BF
2310:41 4C 4C 49 43 41 20 20 A7
2318:20 11 15 43 08 08 08 00 F9
2320:00 00 00 00 00 24 48 3C C3
2328:00 00 00 02 04 02 00 02 B8
2330:02 04 04 04 0A 0A 0A 09 CE
2338:04 04 20 00 00 0F 90 0A EC
2340:07 00 4D 4F 4E 53 54 45 56
2348:52 20 42 41 53 53 41 41 C7
2350:41 08 08 08 00 00 00 00 BA
2358:00 00 18 24 30 00 00 00 65
2360:02 02 02 00 00 00 04 04 74
2368:04 0F 09 04 08 06 05 20 58
2370:00 00 0F 10 0A 01 00 53 41
2378:59 4E 54 48 45 58 20 20 F9
2380:20 20 20 41 21 21 05 08 96
2388:08 00 00 00 00 00 00 30 03
2390:48 3C 00 00 00 00 02 02 01 17
2398:00 00 00 04 04 04 09 09 6A
23A0:07 98 08 98 10 00 00 0F AA
23A8:10 05 01 00 2A 2A 2A 2A D0
23B0:2A 2A 2A 2A 2A 2A 2A 2A F6
23B8:11 11 11 08 08 08 00 00 CE
23C0:00 00 00 00 24 30 3C 00 61
23C8:00 00 02 02 02 00 00 00 7F
23D0:04 04 04 00 00 00 F8 F8 85
23D8:F8 10 00 00 0F 10 FF 00 58
23E0:00 00 00 00 00 00 00 00 27
```

©



When It Comes To Reading Music, Most People Draw A Blank.

If you're not content with your knowledge of music, Wenger has something that will fill in the blanks.

It's called The Music Class.™ An exciting new 5-part software series for kids, adults, beginners, even professionals.

The Music Class is the consummate music teacher. Its simple step-by-step instructions teach everything from the basics to rhythm to note reading—all at your own speed.

And The Music Class even gives advice, points out errors, and applauds correct answers.

Fundamentals: Make sense of those skinny lines with blobs and tails. All the basics from note reading to rhythm. $49.

Rhythm: What is the real difference between a waltz and a polka, ragtime and rock, and more. A comical little guy named Mr. Metro Gnome is your teacher. $49.

Ear Training: How to hear exactly what's happening in a piece of music. You'll never listen to Bon Jovi or Mancini quite the same way again. $49.

Music Symbols: And you thought they were called squiggles, slashes, and dots. Animated graphics and games will teach you up to 80 musical symbols. $39.

Note Reading: Know the difference between an E-Flat eighth note and a B-Flat quarter note. This is where you learn to read the foreign language called music. $39.

Any Apple II or IIGS with 64K memory can run The Music Class.

So order by calling toll free 1-800-843-1337. In Hawaii and Alaska call collect 612-854-9554.

And begin changing that blank stare into an ear-to-ear grin.

The Music Class™

From the Wenger Music Learning Division, Wenger Corp.
1401 East 79th Street, Minneapolis, Minnesota 55420.

ShapeMaker For Apple II

William C. Vergara

Have you ever wished that you could enliven your programs with eye-catching title pages or graphs and charts labeled with descriptive information? Shape tables are very useful for creating such effects on the Apple II. This comprehensive program makes it easy to create and edit shape tables for a variety of purposes. It runs on any Apple II-series computer, under either ProDOS or DOS 3.3.

Many programs can be enhanced by presenting graphics based on custom character fonts or other special shapes. But designing hi-res characters and generating shape tables can be a complex undertaking. "ShapeMaker" provides a simple means of generating and editing shape tables containing characters and shapes—from very small figures to shapes many times larger than standard Apple characters. You can design your own shapes and characters, or you can copy them automatically from existing shape tables and add them to your own customized table.

Using ShapeMaker

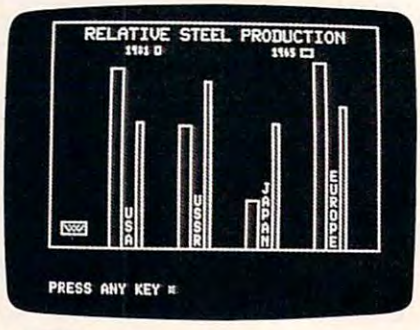
Type in and save Programs 1 and 2. Note that you *must* save Program 2 with the filename SHAPEMAKER because that's the name Program 1 uses when loading Program 2. Program 1 is a very short program that resets the BASIC start-of-program pointer and runs Program 2. This is done to create a protected memory area for hi-res page 1.

Because BASIC memory is modified, you should be *very careful* about editing or resaving either of these programs after you have run them. If you need to edit the program, you should reboot the computer, reload the program from disk, make the desired changes, and resave it before running it again.

When you run the program, it displays a main menu which looks like this:

```
MAIN MENU
(1) DESIGN A NEW SHAPE
(2) ESTABLISH SIZE OF DOT MATRIX
(3) CHANGE STARTING COORDINATES
(4) SAVE SHAPE TABLE TO DISK
(5) LOAD OR START A NEW SHAPE
    TABLE
(6) REVIEW A SHAPE TABLE
(7) COPY SHAPES FROM OLD TO NEW
    TABLE
(8) EDIT THE NEW SHAPE TABLE
(9) LEAVE THE PROGRAM
(PRESS <ESC> KEY TO CATALOG A
 DISK)
```

The main menu gives you access to the program's basic functions. To select an option, simply press the indicated key. For instance, you can press the ESC key to display a catalog of the current disk, or press 9 to exit the program and return to BASIC.



"ShapeMaker" for Apple II computers is a convenient tool for creating and editing shape tables. This screen illustrates just one of the ways that shape tables can be used.

Creating A New Shape

You will usually begin with option 5, which clears the screen and prints a menu with two options. Press N to begin a new table from scratch, or L to load a partially completed table from disk.

If you choose to create a new shape table, the program asks you

to enter the table's capacity—that is, the number of shapes which this table will contain. An Apple shape table can hold as many as 255 shapes. Next, the program prompts you to enter the number of rows and columns for the design matrix and the starting point within the matrix. For instance, say that you want to design a character which is seven pixels (screen dots) wide and nine pixels high; you would enter 7 at the column prompt and 9 at the row prompt. The starting point determines where in the matrix you will begin drawing; the lower left corner of the matrix corresponds to coordinates (1,1).

The design matrix can be as wide as 35 columns and as high as 25 rows. On the screen, it is displayed six times its actual size. You are now ready to design your first shape. If you press Q followed by N, you return to the main menu.

Once the matrix has been set up, you can begin designing a new shape by selecting option 1 from the main menu. The matrix appears immediately with a blinking cursor, which indicates your current position. Beneath the matrix, Shapemaker displays the capacity of the new shape table and the number of the shape being designed.

As indicated by the prompts, you can move the cursor with the arrow keys (for the IIe and IIc) or the keys I, J, K, and M. Press P wherever you want to draw a dot: That point in the matrix is filled. Continue by moving the cursor and plotting pixels until the shape is complete. Pressing Q ends the design process and displays the shape in its true size to the right of the matrix.

At this point you can press Y to add the shape as the next numbered shape in the current table, or press N to discard it. In either case, the program returns to the main menu.

Additional Options

From the main menu, you can select option 2 to change the matrix size or option 3 to select new starting coordinates for the next shape. It's a good idea to save new shape tables frequently to protect against a power failure or other accidents. To save a file, choose option 4 and follow the prompts on the screen. After it saves the file, ShapeMaker prints the starting address and the length of the file in bytes, which you may wish to record for future reference.

ShapeMaker goes to some lengths to protect against mistakes. If you hit the wrong key by mistake, it usually gives you another chance to repeat the input or sends you back to the main menu.

Option 7 allows you to copy an existing shape table into a new table. Again, you can simply follow the prompts on the screen after selecting this option. If there is a source shape table in memory, the program prints its name and asks whether you wish to copy that table. If no table is in memory, ShapeMaker asks you to enter both the name of the desired shape table and the drive where it can be found. At that point, the program gives a description of the source table and asks for the number of the source shape you wish to copy. Before proceeding with the copy, it allows you to verify that this is the correct shape by displaying it on the screen. If you press Y (yes), that shape is copied to the end of the current shape table.

It can take several seconds to copy a shape, so ShapeMaker prints a flashing reminder while it is busy. When the copying is complete, you can either enter the next shape number to copy or enter 0 to exit to the main menu.

Option 6 allows you to review either the source or the destination table.

Option 8 lets you edit the new shape table. When you choose this option, it displays a four-item menu asking whether you wish to insert a shape, delete a shape, increase the table's capacity, or decrease its capacity. For the first two items, you'll need to enter the number of the shape to insert or delete. Inserting shape 7, for instance, has the effect of moving upward all existing shapes with the number 7 and above, and putting in a new shape as number 7.

When you insert, ShapeMaker asks whether you will design the new shape from scratch or copy it from a source table in memory. For a new design, the program jumps to the design matrix. If you choose to copy the new shape into place, the program follows the normal copying procedure, but inserts the shape where indicated rather than adding it to the end of the shape table. If the shape table is full before insertion, the table capacity is increased by one to make room for the inserted shape (as long as the number of shapes would not exceed 255). Deleting a shape removes it from the table and decrements the number of each shape higher than the number chosen.

When you select option 9 (quit), ShapeMaker checks to see if the new table has been changed since the last save. If so, it asks whether you really want to lose the changed shape table, and exits only if you respond with Y (yes).

Custom Character Sets

Programs 3–5 are three sample character sets which you can use immediately or modify further to your own tastes. Each shape table must be entered with the "MLX" machine language entry program listed elsewhere in this issue. Here are the starting and ending addresses needed to enter these files with MLX:

Program 3. SHAPETABLE3X6

STARTING ADDRESS?	7800
ENDING ADDRESS?	7BFF

Program 4. SHAPETABLE5X7

STARTING ADDRESS?	7800
ENDING ADDRESS?	7CA7

Program 5. SHAPETABLE7X9

STARTING ADDRESS?	7800
ENDING ADDRESS?	7F8F

The first shape table (SHAPETABLE3X6) contains 58 uppercase letters, numerals, and other ASCII characters in a format three pixels wide and six pixels high. The characters are small enough so that 70 of them can be placed across the high-resolution screen spaced one pixel apart.

The second table (SHAPETABLE5X7) duplicates the standard Apple character set in size, with uppercase, lowercase, and all other standard characters. The third shape table (SHAPETABLE7X9) includes a larger version of the previous table, plus a complete Greek alphabet.

Once you have saved these files to disk, they can be loaded,

reviewed, and edited at any time with ShapeMaker.

Displaying A Shape Table

Program 6 is a short BASIC program that will display any of the three example shape tables in its entirety. To view a shape table, run Program 6 and answer the two prompts requesting a filename and drive number. The shape table will be displayed on the monitor screen in several rows of 20 characters each.

Hi-Res Bar Chart

Programs 7 and 8 aren't necessary to use ShapeMaker, but you may want to type them in to view an example of what can be done with shape tables. Program 7 is a BASIC program that loads a shape table into memory and uses it to create a bar chart on the hi-res screen. (See photo.) Program 8 is the shape table data for Program 7. It should be entered with MLX using these addresses:

Program 8. BARTABLE

STARTING ADDRESS?	7800
ENDING ADDRESS?	79EF

For Program 7 to function properly, you must save the data from Program 8 with the name BARTABLE. (See line 300 of Program 7.)

Program 1: SHAPEBOOT

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

```
D2 10 REM THIS PGM RESETS THE ST
      ART OF BASIC
95 20 REM IT ALSO RUNS SHAPEMAKE
      R
08 30 POKE 103,1: POKE 104,64: P
      OKE 16384,0: REM PUT BASIC
       ABOVE HIRES PAGE 1
E8 40 PRINT CHR$ (4);"RUN SHAPEM
      AKER"
```

Program 2: SHAPEMAKER

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

```
34 5 GOSUB 3420
5B 10 TEXT : HOME : IF PEEK (103
      ) + 256 * PEEK (104) < > 1
      6385 THEN PRINT : PRINT "R
      UN SHAPEBOOT TO SET START
      OF BASIC": PRINT : GOSUB 2
      120: GOTO 1040
60 20 BL = 0:KT = 0:X = 0:Y = 0:
      I = 0:CODE = 0:M$ = "":Q =
       0:XZ = 0:BH = 0:P = 0:H =
      0:R = 0:C = 0:ADDR = 0
BE 30 ONERR GOTO 1560
50 40 HCOLOR= 3: SCALE= 1: ROT=
      0:TA = 30720:TB = 2304: PO
      KE TB,0: POKE TA,0: REM
      $7800 (NEW) $900 (OLD) TA
      BLES
30 50 TC = 7938:FLAG = 0
F9 60 POKE 768,1: POKE 769,0: PO
      KE 770,4: POKE 771,0: REM
       CURSOR TABLE
3D 70 POKE 772,112: POKE 773,30:
       POKE 774,7: POKE 775,32:
      POKE 776,0
D7 80 POKE TC,1: POKE TC + 1,0:
      POKE TC + 2,4: POKE TC + 3
      ,0: REM  EDIT TABLE
39 90 GOTO 840
BD 100 TEXT : HOME :SN = 0
07 110 PRINT "PLEASE PRESS:": PR
      INT : PRINT "  L  TO LOAD
       A SHAPE TABLE FROM DISK"
      : PRINT "  N  TO START A
      NEW SHAPE TABLE": PRINT :
       PRINT : PRINT "PRESS ANY
       OTHER KEY FOR MAIN MENU"
      : PRINT : PRINT
BB 120 PRINT "YOUR SELECTION: ";
      : GET A$: PRINT A$
AE 130 IF A$ = "N" THEN 220
C8 140 IF A$ < > "L" THEN 840
11 150 PRINT : INPUT "NAME OF TA
      BLE ";N$
B1 160 GOSUB 1540: REM  GET DRIV
      E #
7E 170 PRINT CHR$ (4);"BLOAD ";N
      $;",A";TA;",D";AN
20 180 N = ( PEEK (TA + 2) + 256
       * PEEK (TA + 3) - 2) / 2
4A 190 SN = PEEK (TA): PRINT : I
      F SN = N THEN PRINT "TABL
      E FULL": PRINT : GOSUB 21
      20: GOTO 840
59 200 ADDR = PEEK (TA + SN * 2
      + 2) + PEEK (TA + SN * 2
      + 3) * 256 + TA
B4 210 GOSUB 3400: GOSUB 2120: G
      OTO 840
F1 220 TEXT : HOME : PRINT "PLEA
      SE ENTER THE DESIRED NUMB
      ER": INPUT "OF SHAPES FOR
       THIS TABLE: ";N: IF N >
      255 OR N < 1 THEN PRINT :
      EN = 2: GOTO 1580
BB 230 POKE TA,0: POKE TA + 1,0
CA 240 D1 = 2 * N + 2
A4 250 POKE TA + 2,D1 - 256 * IN
      T (D1 / 256)
22 260 POKE TA + 3, INT (D1 / 25
      6)
25 270 FOR I = TA + 4 TO TA + 2
      * N + 3: POKE I,0: NEXT I
6F 280 PRINT : PRINT "CHOOSE SIZ
      E OF SHAPE DESIGN GRID"
36 290 INPUT "NUMBER OF COLUMNS
      (1 - 35) ";C:C = 6 * C: I
      F C > 210 THEN 290
1E 300 INPUT "NUMBER OF ROWS (1
      - 25) ";R:R = 150 - 6 * R
      : IF R < 0 THEN 300
4F 310 HGR : POKE 250,R: POKE 25
      1,C
49 320 IF C = 0 OR R = 150 THEN
      1550
B9 330 FOR I = R TO 150 STEP 6:
      HPLOT 0,I TO C,I: NEXT I
C5 340 FOR I = 0 TO C STEP 6: HP
      LOT I,R TO I,150: NEXT I
17 350 IF F1 = 1 THEN 370
06 360 IF A = 1 OR FLAG = 1 THEN
       390
71 370 HOME : VTAB 21: PRINT "OR
      IGIN OF SHAPE?  LOWER LEF
      T IS (1,1)"
EB 380 INPUT "COLUMN ";X1: INPUT
       "ROW ";Y1
DB 390 X = 6 * X1 - 3:Y = 153 -
      6 * Y1
5F 400 IF F1 = 1 THEN 2240
C0 410 IF SN = N AND FLAG = 0 TH
      EN PRINT "TABLE IS FILLED
       TO PRESENT CAPACITY": PR
      INT : PRINT : GOSUB 2120:
       GOTO 840
7E 420 IF FLAG = 1 THEN ADDR = P
      EEK (TC + 2) + TC: GOTO 4
      40
7F 430 ADDR = PEEK (TA + SN * 2
      + 2) + 256 * PEEK (TA + S
      N * 2 + 3) + TA
D4 440 POKE 232,0: POKE 233,3: R
      EM  CURSOR TABLE
64 450 HOME : VTAB 21: PRINT "TA
      BLE CAPACITY: ";N;" SHAPE
      S-THIS IS # ";
08 460 IF FLAG = 0 THEN PRINT SN
       + 1: GOTO 480
63 470 PRINT IS
44 480 VTAB 22: PRINT "TO MOVE C
      URSOR, USE IJKM OR ARROW
      KEYS"
25 490 VTAB 23: PRINT "PRESS:  P
       TO PLOT A POINT"
8E 500 VTAB 24: PRINT TAB( 9);"Q
       TO END THIS SHAPE";
90 510 CODE = 0
97 520 GOSUB 1460:H=8:REM DRAW B
      LINKING CURSOR
24 540 IF M$ = "P" THEN CODE = 4
      : FOR I = X - 1 TO X + 1:
       HPLOT I,Y - 1 TO I,Y + 1
      : NEXT I: GOTO 520
27 550 IF M$ = "Q" THEN POKE ADD
      R,CODE: POKE ADDR + 1,255
      : GOTO 660
EB 590 POKE 6, ASC (M$): POKE 8,
      Y: POKE 9,X: POKE 252,COD
      E: CALL 2048
FE 600 H = PEEK (7):X = PEEK (9)
      :Y = PEEK (8)
4E 610 IF H = 8 THEN 520
78 640 POKE ADDR,H:ADDR = ADDR +
      1
9C 650 GOTO 510
78 660 IF FLAG = 1 THEN ADDR = P
      EEK (TC + 2) + TC:LOC = A
      DDR: GOTO 680
94 670 ADDR = PEEK (TA + SN * 2
      + 2) + 256 * PEEK (TA + S
      N * 2 + 3) + TA:LOC = ADD
      R
B0 680 V1 = PEEK (LOC): IF V1 =
      255 THEN POKE ADDR,0:ADDR
       = ADDR + 1: GOTO 780
EF 690 V2 = PEEK (LOC + 1): IF V
      2 = 255 THEN POKE ADDR,V1
      : POKE ADDR + 1,0:ADDR =
      ADDR + 2: GOTO 780
DE 700 V3 = PEEK (LOC + 2): IF V
      3 = 255 THEN POKE ADDR,V1
       + 8 * V2: POKE ADDR + 1,
      0:ADDR = ADDR + 2: GOTO 7
      80
EF 710 BYTE = V1 + 8 * V2 + 64 *
       V3
A4 720 IF BYTE = 0 THEN POKE ADD
      R,64: POKE ADDR + 1,24:AD
      DR = ADDR + 2:LOC = LOC +
       3: GOTO 680: REM  USES 3
       SKIP-UP VECTORS
7D 730 IF V3 > 0 AND V3 < 4 THEN
       POKE ADDR,BYTE:ADDR = AD
      DR + 1:LOC = LOC + 3: GOT
      O 680: REM  USE ALL 3 VEC
      TORS
32 740 BYTE = V1 + 8 * V2: REM
      3RD VECTOR NOT USED FROM
      HERE ON
06 750 IF V2 > 0 THEN POKE ADDR,
      BYTE:ADDR = ADDR + 1:LOC
```

```
      = LOC + 2: GOTO 680: REM
       2 VECTORS USED
F9 760 IF V1 < > 0 THEN POKE ADD
      R,BYTE:ADDR = ADDR + 1:LO
      C = LOC + 1: GOTO 680: RE
      M  1 VECTOR USED
66 770 IF BYTE = 0 THEN POKE ADD
      R,24: POKE ADDR + 1,8:ADD
      R = ADDR + 2:LOC = LOC +
      2: GOTO 680: REM  2 SKIP-
      UPS AND 2 OFFSETTING MOVE
      S SIDEWAYS
FE 780 IF FLAG = 1 THEN POKE 232
      ,TC - INT (TC / 256) * 25
      6: POKE 233, INT (TC / 25
      6): DRAW 1 AT 200,100: GO
      TO 2270
B4 790 POKE TA,SN + 1: POKE 232,
      TA - INT (TA / 256) * 256
      : POKE 233, INT (TA / 256
      ): XDRAW SN + 1 AT 245,10
      0
B4 800 HOME : PRINT : VTAB 22: P
      RINT "SAVE THIS AS SHAPE
      NUMBER ";SN + 1;"(Y/N)";:
       GET A$
D8 810 IF A$ = "Y" THEN SN = SN
      + 1:SF = 1: IF SN < N THE
      N D1 = ADDR - TA: POKE TA
       + 2 * SN + 2,D1 - 256 *
      INT (D1 / 256): POKE TA +
       2 * SN + 3, INT (D1 / 25
      6)
48 820 IF A$ < > "N" AND A$ < >
      "Y" THEN 800
9C 830 POKE TA,SN
41 840 TEXT : HOME : PRINT "  SH
      APEMAKER"; TAB( 32);"MAIN
       MENU": VTAB 4: PRINT "PL
      EASE MAKE A SELECTION:":
      PRINT : PRINT
65 850 FLAG = 0:EX = 0:BL = FRE
      (0)
01 860 PRINT TAB( 3);"(1) DESIGN
       A NEW SHAPE"
91 870 PRINT TAB( 3);"(2) ESTABL
      ISH SIZE OF DOT MATRIX"
EA 880 PRINT TAB( 3);"(3) CHANGE
       STARTING COORDINATES"
87 890 PRINT TAB( 3);"(4) SAVE S
      HAPE TABLE TO DISK"
1D 900 PRINT TAB( 3);"(5) LOAD O
      R START A NEW SHAPE TABLE
      "
F9 910 PRINT TAB( 3);"(6) REVIEW
       A SHAPE TABLE"
7C 920 PRINT TAB( 3);"(7) COPY S
      HAPES FROM OLD TO NEW TAB
      LE"
6B 930 PRINT TAB( 3);"(8) EDIT T
      HE NEW SHAPE TABLE"
0F 940 PRINT TAB( 3);"(9) LEAVE
      THE PROGRAM"
BF 950 PRINT : PRINT : PRINT "(P
      RESS <ESC> KEY TO CATALOG
       A DISK)"
64 960 PRINT : PRINT : PRINT "YO
      UR SELECTION: ";: GET A$:
       PRINT A$:A = VAL (A$)
57 970 IF ASC (A$) = 27 THEN 143
      0
3F 980 IF A < 1 OR A > 9 THEN 84
      0
AF 990 PRINT : IF N = 0 AND A <
      5 THEN PRINT "NO TABLE AV
      AILABLE.  LOAD OR INITIAL
      IZE": PRINT "A SHAPE TABL
      E BEFORE DESIGNING SHAPES
      .": PRINT : GOSUB 2120: G
      OTO 840
80 1000 IF SN = 255 AND A < 4 TH
      EN : HOME : PRINT "THE S
      HAPE TABLE IS FULL": PRI
      NT : GOSUB 2120: GOTO 84
      0
AA 1010 ON A GOTO 310,280,310,10
      50,1370,1140,1700,2125,1
      020
FA 1020 IF SF = 0 THEN 1040
FB 1030 HOME : PRINT "YOU ARE AB
      OUT TO PERMANENTLY LOSE"
      ; CHR$ (7): PRINT "THE S
      HAPE TABLE FILE IN MEMOR
      Y": PRINT "DO YOU REALLY
       WANT TO DO THAT? (Y/N)
      ";: GET AN$: IF AN$ < >
      "Y" THEN 840
C0 1040 TEXT : HOME : POKE 103,1
      : POKE 104,8: POKE 2048,
      0: POKE 2049,0: POKE 205
      0,0: VTAB 7: PRINT "THE
      APPLESOFT POINTER HAS NO
      W BEEN RESET": PRINT "TO
       ITS NORMAL LOCATION IN
      MEMORY.": VTAB 11: PRINT
       "TO LIST THIS PROGRAM F
      IRST USE COMMAND
8A 1045 VTAB 14: HTAB 12: PRINT
      "LOAD SHAPEMAKER": END
9C 1050 INPUT "NAME ";N$
77 1060 GOSUB 1540
E6 1070 BL = PEEK (TA + SN * 2)
      + 256 * PEEK (TA + SN *
      2 + 1) + TA
72 1080 FOR EO = BL TO BL + 2000
      : IF PEEK (EO) = 0 THEN
      L = EO - TA + 2: GOTO 11
      00
2B 1090 NEXT EO
F6 1100 PRINT CHR$ (4);"BSAVE ";
      N$;",A";TA;",L";L;",D";A
      N
D3 1110 PRINT : PRINT "FILE : ";
      N$: PRINT "SAVED AT: ";T
      A;" DECIMAL": PRINT "FIL
      E LENGTH: ";L;" DECIMAL"
      : PRINT : PRINT "PRESS A
       KEY ": GET BL$
0D 1120 SF = 0: GOTO 840
5B 1130 TEXT : HOME : END
B2 1140 IF PEEK (TB) = 0 AND PEE
      K (TA) = 0 THEN PRINT "T
      HERE ARE NO TABLES IN ME
      MORY": PRINT : GOSUB 212
      0: GOTO 840
E7 1150 IF PEEK (TA) = 0 THEN 12
      10
DD 1160 IF PEEK (TB) = 0 THEN 11
      90
30 1170 HOME : PRINT "WHICH SHAP
      E TABLE DO YOU WANT TO S
      EE?": PRINT : PRINT "PRE
      SS:": HTAB 3: PRINT "1 F
      OR THE TARGET (NEW) TABL
      E": HTAB 3: PRINT "2 FOR
       THE SOURCE (OLD) TABLE
      ": GET AN$
E2 1180 IF AN$ < > "1" THEN 1200
7F 1190 TT = TA:SS = SN: GOTO 12
      20
EA 1200 IF AN$ < > "2" THEN 840
16 1210 TT = TB:SS = OS
53 1220 POKE 232,TT - INT (TT /
      256) * 256: POKE 233, IN
      T (TT / 256)
F5 1230 HOME : PRINT "THERE ARE
      ";SS;" SHAPE(S) IN THE T
      ABLE"
C8 1240 NN = ( PEEK (TT + 2) + 2
      56 * PEEK (TT + 3) - 2)
      / 2: PRINT "TABLE CAPACI
      TY IS ";NN;" SHAPES"
04 1250 IF SS = 0 THEN PRINT : G
      OSUB 2120: GOTO 840
64 1260 PRINT : PRINT "ENTER 0 T
      O RETURN TO MAIN MENU":
      PRINT
61 1270 INPUT "OR ENTER NUMBER O
      F DESIRED SHAPE ";DS$:DS
       = VAL (DS$): IF DS > SS
       THEN 1230
38 1280 IF DS = 0 THEN 840
6F 1290 POKE 232,TT - INT (TT /
      256) * 256: POKE 233, IN
      T (TT / 256)
9B 1300 HGR
C4 1310 XDRAW DS AT 220,100: VTA
      B 21: CALL - 868: PRINT
      "CURRENT SHAPE IS # ";DS
B8 1320 VTAB 22: PRINT "ENTER NU
      MBER OF NEXT DESIRED SHA
      PE,": CALL - 868: INPUT
      "OR ENTER 0 TO RETURN TO
       MAIN MENU  ";AN$: IF AN
      $ = "0" THEN 840
C7 1330 XDRAW DS AT 220,100:DS =
       VAL (AN$)
B0 1340 IF DS > SS OR DS < 1 THE
      N TEXT : PRINT CHR$ (7):
       GOTO 1230
2E 1350 IF DS = 0 THEN 840
72 1360 GOTO 1310
5B 1370 HOME : IF SF = 0 THEN 10
      0
5F 1380 PRINT "THE SHAPE TABLE I
      N MEMORY WILL BE": PRINT
       "LOST IF YOU START A NE
      W SHAPE": PRINT "TABLE.
       DO YOU REALLY WANT TO D
      O": PRINT "THAT? (Y/N) "
      ; CHR$ (7);
EB 1390 GET AN$
7E 1400 IF AN$ = "Y" THEN 100
13 1410 IF AN$ < > "Y" AND AN$ <
       > "N" THEN PRINT : PRIN
      T : GOTO 1380
E3 1420 GOTO 840
77 1430 GOSUB 1540
DB 1440 PRINT CHR$ (4);"CATALOG,
      D";AN
CA 1450 PRINT : GOSUB 2120: GOTO
       840
D2 1460 XZ = PEEK (49168):KT = 0
15 1470 XDRAW 1 AT X,Y:KT = KT +
       1
C5 1480 IF KT = 2 THEN KT = 0
4C 1490 FOR I = 1 TO 30:BL = PEE
      K (49152): IF BL > 127 T
      HEN M$ = CHR$ (BL - 128)
      :BL = 0: GOTO 1520
75 1500 NEXT I
7C 1510 GOTO 1470
2A 1520 IF KT = 1 THEN XDRAW 1 A
      T X,Y
E7 1530 RETURN
F1 1540 HOME : PRINT "ENTER DISK
       DRIVE NUMBER: ";: GET A
      N$: PRINT AN$:AN = VAL (
      AN$): RETURN
C9 1550 TEXT : HOME : PRINT "THE
      RE'S NO SHAPE DESIGN MAT
      RIX IN MEMORY": PRINT "P
      LEASE ESTABLISH ONE": PR
      INT : GOSUB 2120: GOTO 8
      40
B4 1560 TEXT : HOME : PRINT : PR
      INT "OOPS! CAN'T DO THAT
      ": PRINT CHR$ (7)
5B 1570 EN = PEEK (222)
51 1580 IF EN = 2 OR EN = 3 THEN
       PRINT "THAT NUMBER IS T
      OO BIG OR SMALL"
E4 1590 IF EN = 4 THEN PRINT "SO
      RRY, CAN'T WRITE TO A WR
      ITE": PRINT "PROTECTED F
      ILE"
4C 1600 IF EN = 6 THEN PRINT "SO
```

```
      RRY, CAN'T FIND THAT FIL
      E"
79 1610 IF EN = 8 THEN PRINT "TH
      ERE'S SOME SORT OF INPUT
      /OUTPUT": PRINT "ERROR"
03 1620 IF EN = 9 THEN PRINT "SO
      RRY, THAT DISK IS ALREAD
      Y FULL OF DATA"
F8 1630 IF EN = 10 THEN PRINT "S
      ORRY, CAN'T WRITE TO A L
      OCKED FILE"
2A 1640 IF EN = 11 OR EN = 16 TH
      EN PRINT "THERE'S SOME S
      ORT OF SYNTAX ERROR HERE
      "
D6 1650 IF EN = 53 THEN PRINT "S
      ORRY, THAT NUMBER IS NOT
       LEGAL"
38 1660 IF EN = 77 THEN PRINT "O
      H OH!  WE'RE OUT OF MEMO
      RY!"
2A 1670 ONERR GOTO 1560
B8 1680 PRINT : PRINT "LET'S RET
      URN TO THE MAIN MENU AND
      ": PRINT "TRY AGAIN": PR
      INT : GOSUB 2120: GOTO 8
      40
86 1690 OT$ = "": GOTO 1560
69 1700 IF SN = N THEN PRINT "TA
      BLE IS FULL": GOSUB 2120
      : GOTO 840
DB 1710 HOME : IF OT$ < > "" THE
      N PRINT "THE SOURCE SHAP
      E TABLE IN MEMORY IS:":
      PRINT OT$: PRINT : PRINT
       "IS THAT OK? (Y/N):": G
      ET AN$: IF AN$ = "Y" THE
      N 1760
A2 1720 PRINT : PRINT "ENTER NAM
      E OF SOURCE SHAPE TABLE:
       ": INPUT "";OT$:
79 1730 GOSUB 1540
E1 1740 ONERR GOTO 1690
09 1750 PRINT CHR$ (4);"BLOAD ";
      OT$;",A";TB;",D";AN
BB 1760 HOME : IF N = 0 THEN PRI
      NT "THERE IS NO TARGET T
      ABLE AVAILABLE IN": PRIN
      T "MEMORY.  PLEASE INITI
      ALIZE A NEW TABLE": PRIN
      T "OR LOAD ONE FROM MEMO
      RY": PRINT : PRINT "RETU
      RNING TO MAIN MENU": PRI
      NT : GOSUB 2120: GOTO 84
      0
DB 1770 NO = ( PEEK (TB + 2) + 2
      56 * PEEK (TB + 3) - 2)
      / 2
AE 1780 OS = PEEK (TB)
D8 1790 OA = PEEK (TB + OS * 2 +
       2) + PEEK (TB + OS * 2
      + 3) * 256 + TB: REM  AD
      DRESS OF OLD TABLE
6F 1800 PRINT "THE SOURCE TABLE
      CAN HOLD ";NO;" SHAPES":
       PRINT : PRINT "IT NOW H
      AS ";OS;" SHAPES IN IT":
       PRINT : GOSUB 2120:
55 1810 HOME : HGR : VTAB 24: PR
      INT "ENTER 0 TO RETURN T
      O MAIN MENU"
4A 1820 VTAB 21: HTAB 36: PRINT
      "  "
31 1830 VTAB 21: HTAB 1: PRINT "
      ENTER SOURCE SHAPE NUMBE
      R TO COPY: ";: INPUT "";
      AN$:AN = VAL (AN$)
F0 1840 IF AN = 0 THEN 840
A8 1850 IF AN < 1 OR AN > OS THE
      N PRINT "SORRY, NO SUCH
      SHAPE IN TABLE": PRINT "
      PLEASE PRESS A KEY ": GE
      T AN$: GOTO 1810
EA 1860 POKE 232,TB - INT (TB /
      256) * 256: POKE 233, IN
      T (TB / 256)
1E 1870 DRAW AN AT 200,100: IF F
      LAG = 1 THEN 2630
65 1880 HOME : VTAB 21: PRINT "C
      OPY THIS AS SHAPE #";SN
      + 1;" (Y/N) ";: GET AN$
B5 1890 IF AN$ = "N" THEN 1810
6B 1900 IF AN$ < > "Y" THEN 840
A7 1910 VTAB 24: HTAB 1: FLASH :
       PRINT "COPYING DATA";:
      NORMAL : PRINT "
B6 1920 ADDR = PEEK (TA + SN * 2
      + 2) + 256 * PEEK (TA +
      SN * 2 + 3) + TA
16 1930 SF = 1: IF SN < N THEN D
      1 = ADDR - TA: POKE TA +
      2 * SN + 2,D1 - 256 * I
      NT (D1 / 256): POKE TA +
       2 * SN + 3, INT (D1 / 2
      56)
D2 1940 POKE TA,SN + 1
A5 1950 OA = PEEK (TB + 2 * AN)
      + PEEK (TB + 2 * AN + 1)
       * 256 + TB
31 1960 Z1 = 0
32 1970 FOR I = 1 TO 1000
F4 1980 POKE 232,TB - INT (TB /
      256) * 256: POKE 233, IN
      T (TB / 256)
34 1990 BI = PEEK (OA)
83 2000 POKE 232,TA - INT (TA /
      256) * 256: POKE 233, IN
      T (TA / 256)
C7 2010 POKE ADDR,BI
3D 2020 ADDR = ADDR + 1
D4 2030 OA = OA + 1
D5 2040 IF BI = 0 AND Z1 = 0 THE
      N POKE ADDR,0:ADDR = ADD
      R + 1: GOTO 2080
B9 2050 IF BI = 0 THEN 2080
05 2060 Z1 = Z1 + 1
B8 2070 NEXT I
CB 2080 SN = SN + 1: IF SN < N T
      HEN D1 = ADDR - TA: POKE
       TA + 2 * SN + 2,D1 - 25
      6 * INT (D1 / 256): POKE
       TA + 2 * SN + 3, INT (D
      1 / 256)
7A 2090 VTAB 23: HTAB 1: PRINT "
      DONE, PLEASE PRESS A KEY
      ";: GET AN$
6B 2100 IF SN = N THEN HOME : VT
      AB 23: GOTO 1700
65 2110 GOTO 1810
43 2120 PRINT "PRESS ANY KEY TO
      CONTINUE ";: GET AN$: RE
      TURN
5F 2125 IF N = 0 THEN 1760
4F 2130 HOME : PRINT "PRESS:": P
      RINT : PRINT TAB( 3);"(1
      ) TO INSERT A SHAPE IN T
      ABLE": PRINT TAB( 3);"(2
      ) TO DELETE A SHAPE FROM
       TABLE"
66 2140 PRINT TAB( 3);"(3) TO IN
      CREASE TABLE CAPACITY":
      PRINT TAB( 3);"(4) TO DE
      CREASE TABLE CAPACITY ";
      : GET AN$: PRINT AN$: IF
       AN$ < "1" OR AN$ > "4"
      THEN 840
3B 2150 AN = VAL (AN$)
5F 2160 PL = 0: IF AN = 3 THEN P
      L = 1
66 2170 IF AN = 1 AND SN = 255 T
      HEN PRINT : PRINT "SORRY
      , THE NEW SHAPE TABLE IS
       FULL": PRINT : PRINT "D
      ELETE A SHAPE BEFORE ADD
      ING TO TABLE": GOSUB 212
      0: GOTO 840
48 2180 ON AN GOTO 2190,2740,302
      5,3025
11 2190 HOME : PRINT "ENTER NUMB
      ER OF SHAPE TO BE INSERT
      ED": INPUT "INTO THE NEW
       TABLE ";IS: IF IS < 1 O
      R IS > SN THEN PRINT : P
      RINT "THAT NUMBER IS OUT
       OF RANGE": PRINT : GOSU
      B 2120: GOTO 840
03 2200 FLAG = 1:S1 = SN:NN = N
93 2210 HOME : PRINT "PRESS:": P
      RINT : PRINT TAB( 3);"(1
      ) TO DESIGN NEW SHAPE #"
      ;IS: PRINT TAB( 3);"(2)
      TO GET IT FROM THE SOURC
      E TABLE ";: GET AN$: PRI
      NT AN$: IF AN$ < "1" OR
      AN$ > "2" THEN 2210
31 2220 AN = VAL (AN$)
E0 2230 ON AN GOTO 2240,2610
9C 2240 F1 = 0: IF C = 0 OR R =
      150 THEN F1 = 1: GOTO 28
      0
E9 2250 IF N = 0 THEN PRINT : PR
      INT "NO TARGET TABLE AVA
      ILABLE": PRINT : GOSUB 2
      120: GOTO 840
E5 2260 GOTO 310
5D 2270 HOME : POKE - 16368,0: V
      TAB 22: HTAB 1: PRINT "I
      NSERT THIS AS SHAPE NUMB
      ER ";IS;" (Y/N)?";: GET
      AN$
FD 2280 IF AN$ < > "Y" THEN TEXT
       : HOME : GOTO 2130
D3 2290 HOME : VTAB 24: FLASH :
      PRINT "INSERTING SHAPE N
      UMBER ";IS: NORMAL :SF =
       1
B0 2300 NL = 0
07 2310 FOR I = TC + 4 TO TC + 1
      000
91 2320 BL = PEEK (I):NL = NL +
      1: IF BL = 0 AND NL > 1
      THEN 2350
17 2330 IF BL = 0 THEN NL = 2: G
      OTO 2350
82 2340 NEXT I
74 2350 IF SN = N THEN EX = 2
7A 2360 GOSUB 3340
DE 2370 START = PEEK (TA + IS *
      2) + 256 * PEEK (TA + IS
       * 2 + 1) + TA: REM   FR
      OM IS TO END
58 2380 FOR I = OE TO START STEP
       - 1
26 2390 POKE I + NL + EX, PEEK (
      I)
74 2400 NEXT I
F9 2410 Z = 0
1C 2420 FOR I = TC + 4 TO TC + 4
       + NL - 1: REM  INSERT N
      EW SHAPE
D9 2430 POKE START + EX + Z, PEE
      K (I):Z = Z + 1
84 2440 NEXT I
F4 2450 BEGIN = PEEK (TA + 2) +
      256 * PEEK (TA + 3) + TA
      : REM  ADDRS OF #1 SHAPE
7C 2460 FOR I = START - 1 TO BEG
      IN STEP - 1
27 2470 POKE I + EX, PEEK (I)
94 2480 NEXT I
95 2490 BH = 0
4B 2500 A1 = 2: IF SN + 1 > = N
      THEN A1 = 0
20 2510 FOR I = TA + 2 * SN + A1
       TO TA + 2 * IS STEP - 2
A9 2520 BH = PEEK (I + 1):BL = P
      EEK (I):BL = BL + NL + E
```

```
      X: IF BL > 255 THEN BL =
       BL - 256:BH = BH + 1: P
      OKE I + 3,BH
D0 2530 POKE I + 2,BL: POKE I +
      3,BH
86 2540 NEXT I
1D 2550 FOR I = TA + 2 * IS TO T
      A + 2 STEP - 2
79 2560 BH = PEEK (I + 1):BL = P
      EEK (I):BL = BL + EX: IF
       BL > 255 THEN BL = BL -
       256:BH = BH + 1: POKE I
       + 1,BH
7E 2570 POKE I,BL: POKE I + 1,BH
96 2580 NEXT I
A3 2590 SN = S1 + 1:N = NN + EX
      / 2: POKE TA,SN:D1 = 2 *
       N + 2
A8 2600 HOME : VTAB 23: PRINT "I
      NSERTION OF SHAPE NUMBER
       ";IS;" COMPLETE";: PRIN
      T : GOSUB 2120: GOTO 840
9B 2610 IF N = 0 THEN 2250
71 2620 GOTO 1710
76 2630 HOME : VTAB 21: PRINT "I
      NSERT THIS AS SHAPE # ";
      IS;" (Y/N)";: GET A$: IF
       A$ = "N" THEN 1810
B6 2640 IF A$ < > "Y" THEN 840
1F 2650 OA = PEEK (TB + 2 * AN)
      + PEEK (TB + 2 * AN + 1)
       * 256 + TB: REM  ADDR O
      F SHAPE
12 2660 Z = 0
C7 2670 FOR I = 0 TO 1000: REM
      PUT IT IN TABLE
3A 2680 BI = PEEK (OA + I)
64 2690 POKE TC + 4 + I,BI
1D 2700 IF BI = 0 AND Z > 0 THEN
       2730
75 2710 IF BI = 0 THEN POKE TC +
       4 + I + 1,0: GOTO 2730
82 2720 NEXT I
8E 2730 GOTO 2290
83 2740 PRINT : PRINT "ENTER NUM
      BER OF SHAPE TO DELETE F
      ROM THE": INPUT "TARGET
      (NEW) SHAPE TABLE ";AN$:
      DS = VAL (AN$): IF DS <
      1 OR DS > SN THEN PRINT
      : PRINT "NO SUCH NUMBER
      IN TABLE": PRINT : GOSUB
       2120: GOTO 840
A5 2750 POKE 232,TA - INT (TA /
      256) * 256: POKE 233, IN
      T (TA / 256)
B5 2760 HGR : DRAW DS AT 200,100
60 2770 HOME : VTAB 22: PRINT "D
      ELETE THIS AS SHAPE # ";
      DS;"? (Y/N) ";: GET AN$:
       IF AN$ < > "Y" AND AN$
      < > "N" THEN 840
96 2780 IF AN$ = "N" THEN 2740
EF 2790 HOME : VTAB 22: FLASH :
      PRINT "DELETING SHAPE #
      ";DS: NORMAL :SF = 1
62 2800 ADDR = PEEK (TA + SN * 2
      ) + 256 * PEEK (TA + SN
      * 2 + 1) + TA
E9 2810 IF PEEK (ADDR) = 0 THEN
      EO = ADDR + 1: GOTO 2850
59 2820 FOR EO = ADDR TO ADDR +
      1000: REM  FIND END OF T
      ABLE
B3 2830 BL = PEEK (EO): IF BL =
      0 THEN 2850
20 2840 NEXT EO
F4 2850 START = PEEK (TA + 2 * D
      S) + 256 * PEEK (TA + 2
      * DS + 1) + TA
16 2860 Z = 0
DB 2870 FOR I = START TO START +
       1000: REM  FIND LENGTH
      OF DELETE SHAPE
67 2880 BL = PEEK (I):Z = Z + 1:
       IF BL = 0 THEN 2900
A0 2890 NEXT I
37 2900 IF Z = 1 THEN Z = 2
FE 2910 FOR I = START + Z TO EO:
      REM  MOVE VECTORS
49 2920 POKE I - Z, PEEK (I)
8A 2930 NEXT I
18 2940 EX = 0: IF SN < N THEN E
      X = 2
8A 2950 FOR I = TA + 2 * DS + 2
      TO TA + 2 * SN + EX STEP
       2
56 2960 BH = PEEK (I + 1):BL = P
      EEK (I):BL = BL - Z: IF
      BL < 0 THEN BL = BL + 25
      6:BH = BH - 1
CC 2970 POKE I - 2,BL: POKE I -
      1,BH
9E 2980 NEXT I
24 2990 IF SN = N THEN BL = EO -
       Z + 1: POKE TA + 2 * SN
      ,BL - INT (BL / 256) * 2
      56: POKE TA + 2 * SN + 1
      , INT ((BL - TA) / 256)
B6 3000 IF EX = 2 THEN POKE TA +
       2 * SN + 2,0: POKE TA +
       2 * SN + 3,0
73 3010 SN = SN - 1: POKE TA,SN:
      EX = 0:ADDR = EO + 1
43 3020 HOME : VTAB 22: PRINT "D
      ELETION OF SHAPE NUMBER
      ";DS;" COMPLETED": PRINT
       : GOSUB 2120: GOTO 840
DC 3025 GOSUB 3400: IF PL = 0 TH
      EN PRINT "SUBTRACT ";: G
      OTO 3035
81 3030 PRINT "ADD ";
14 3035 INPUT "HOW MANY SHAPES?
      ";NC$:NC = VAL (NC$)
6D 3040 GOSUB 3340
70 3050 START = PEEK (TA + 2) +
      256 * PEEK (TA + 3) + TA
CF 3060 IF PL = 0 THEN 3230
8B 3070 IF N + NC > 255 THEN HOM
      E : PRINT "TOO MANY SHAP
      ES!": PRINT : GOSUB 2120
      : GOTO 840
3A 3080 GOSUB 3410:SF = 1
57 3090 FOR I = OE TO START STEP
       - 1
0E 3100 POKE I + 2 * NC, PEEK (I
      )
73 3110 NEXT I
6E 3120 FOR I = START TO TA + 2
      * (N + NC) + 1
2E 3130 POKE I,0
7F 3140 NEXT I
EF 3150 FOR I = TA + 2 TO START
       - 2 STEP 2
07 3160 BL = PEEK (I):BH = PEEK
      (I + 1): IF BL = 0 AND B
      H = 0 THEN 3210
BB 3170 BL = BL + 2 * NC
0D 3180 IF BL > 255 THEN BL = BL
       - 256:BH = BH + 1: GOTO
      3180
7F 3190 POKE I,BL: POKE I + 1,BH
71 3200 NEXT I
E7 3210 BH = INT ((OE - TA + 2 *
      NC + 1) / 256):BL = OE
      + 2 * NC + 1 - TA - BH *
      256: POKE TA + SN * 2 +
      2,BL: POKE TA + SN * 2
      + 3,BH:N = N + NC
21 3220 GOSUB 3400: GOSUB 2120:
      GOTO 840
C5 3230 IF N - NC < 1 OR N - NC
      > 255 THEN 2130
04 3235 IF NC < 1 THEN EN = 2: G
      OTO 1580
44 3240 IF SN > N - NC THEN PRIN
      T "DO YOU WANT TO LOSE "
      ;SN - N + NC;" SHAPE(S)?
       (Y/N) ";: GET AN$: IF A
      N$ < > "Y" THEN 840
32 3250 GOSUB 3410:SF = 1
5C 3260 FOR I = START TO OE: POK
      E I - 2 * NC, PEEK (I):
      NEXT I
B6 3270 FOR I = TA + 2 TO START
      - 2 - 2 * NC STEP 2
D1 3280 BL = PEEK (I):BH = PEEK
      (I + 1): IF BL = 0 AND B
      H = 0 THEN 3320
C9 3290 BL = BL - 2 * NC
92 3300 IF BL < 0 THEN BL = BL +
       256:BH = BH - 1: GOTO 3
      300
C1 3310 POKE I,BL: POKE I + 1,BH
      : NEXT I
09 3320 N = N - NC: IF SN > N TH
      EN SN = N: POKE TA,SN
6C 3330 GOTO 3220
C1 3340 Z = 0:ADDR = PEEK (TA +
      SN * 2) + 256 * PEEK (TA
       + SN * 2 + 1) + TA
20 3350 FOR I = 1 TO 1000
9E 3360 BL = PEEK (ADDR): IF BL
      = 0 AND Z > 0 THEN 3390
9A 3370 IF BL = 0 THEN ADDR = AD
      DR + 1: GOTO 3390
8B 3380 ADDR = ADDR + 1:Z = Z +
      1: NEXT I
73 3390 OE = ADDR: RETURN
6B 3400 HOME : PRINT "THIS TABLE
       CAN HOLD ";N;" SHAPES":
       PRINT : PRINT "IT NOW H
      AS ";SN;" SHAPE(S) IN IT
      ": PRINT : PRINT : RETUR
      N
9E 3410 PRINT : FLASH : PRINT "A
      DJUSTING CAPACITY": NORM
      AL : RETURN
1F 3420 FOR I = 1 TO 178: READ X
      : POKE 2047 + I,X: NEXT
      I: RETURN
AE 3430 DATA 165,6,201,73,240,04
      ,201,11,208,37,165,252,1
      33,7,56,165,8,201,3,240,
      16,233,6,133,8,197,250,1
      44,3
60 3440 DATA 76,177,8,24,105,6,1
      33,8,32,58,255,169,8,133
      ,7,76,177,8,201,75,240,4
      ,201,21,208,37,24
F3 3450 DATA 165,252,105,1,133,7
      ,24,165,9,105,6,133,9,16
      5,251,197,9,176,102,56,1
      65,9,233,6
9E 3460 DATA 133,9,32,58,255,169
      ,8,133,7,76,177,8,201,77
      ,240,4,201,10,208,37,24,
      165,252,105,2,133,7
AB 3470 DATA 24,165,8,105,6,133,
      8,169,150,197,8,176,57,5
      6,165,8,233,6,133,8,32,5
      8,255,169,8
CE 3480 DATA 133,7,76,177,8,201,
      74,240,4,201,8,208,32,24
      ,165,252,105,3,133,7,56,
      165,9,233,6
AD 3490 DATA 133,9,165,9,201,240
      ,144,12,24,105,6,133,9,3
      2,58,255,169,8,133,7,96
53 50000 D$ = CHR$ (4):I$ = CHR$
       (9)
1E 50020 PRINT D$;"PR#1"
2F 50040 PRINT I$;"60P"
B5 50050 PRINT I$;"80N"
6E 50060 LIST 1,4000
44 50070 PRINT D$;"PR#0"
```

Program 3: SHAPETABLE3X6

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

```
7800: 3A 00 76 00 7C 00 82 00 C5
7808: 8E 00 98 00 A0 00 A8 00 A9
7810: AC 00 B4 00 BB 00 C1 00 4F
7818: C6 00 CA 00 CE 00 D1 00 DF
7820: D7 00 DF 00 E6 00 EF 00 10
7828: F8 00 FF 00 06 01 0E 01 E6
7830: 15 01 1F 01 26 01 29 01 68
7838: 2E 01 34 01 39 01 40 01 66
7840: 47 01 49 01 51 01 59 01 90
7848: 61 01 6A 01 74 01 7B 01 27
7850: 85 01 8E 01 98 01 A0 01 31
7858: A9 01 B0 01 BA 01 C4 01 E8
7860: CD 01 D4 01 DE 01 E7 01 EE
7868: EE 01 F5 01 FE 01 07 02 EB
7870: 10 02 18 02 1E 02 12 04 2E
7878: 20 24 04 00 40 18 33 0D D8
7880: 24 00 21 24 17 27 17 15 82
7888: 36 77 6E 24 27 00 24 BC C0
7890: 6E B0 02 30 1E 3C 04 00 12
7898: 2C 20 9F 8D 12 1F 26 00 CD
78A0: 36 E5 23 05 20 E5 04 00 7B
78A8: 40 03 24 00 23 C5 0C 96 DD
78B0: 1A 76 04 00 24 1C 8E 12 8D
78B8: F6 04 00 E0 0D 16 1F 04 37
78C0: 00 24 15 1F 04 00 32 1E F1
78C8: 04 00 38 0D 04 00 12 04 DB
78D0: 00 2C 20 D7 92 26 00 21 9C
78D8: E4 1E 36 76 05 20 00 20 BA
78E0: BC 95 36 6F 04 00 21 E4 9A
78E8: 17 8E D1 17 2D 04 00 28 56
78F0: 20 3F 8E 11 36 3F 04 00 5B
78F8: 38 24 0D 36 36 26 00 38 96
7900: 24 2D 96 F6 27 00 38 64 A0
7908: 95 32 1E 1C 24 00 18 20 49
7910: 2D 36 36 26 00 38 28 70 F1
7918: D7 32 0E 05 20 04 00 38 DE
7920: 24 2D 36 36 26 00 04 20 F3
7928: 00 04 B0 13 04 00 15 C7 75
7930: 28 28 20 00 25 38 B7 04 C2
7938: 00 1A 0C 0C 1C 1C 04 00 4D
7940: C0 2C 35 F6 16 04 00 00 75
7948: 00 25 E4 1E 36 36 0D 24 CB
7950: 00 60 1C 37 36 36 65 24 CB
7958: 00 23 24 2D B6 32 3F 24 8C
7960: 00 21 3C 38 36 36 2E 05 92
7968: 20 00 08 40 03 3F 36 F5 E7
7970: 36 2D 04 00 60 38 37 36 D2
7978: 36 04 00 08 40 03 3F 36 CA
7980: 36 2E 25 24 00 28 24 1F 09
7988: 36 36 6E 24 04 00 24 3C D8
7990: 0D 96 1A 36 08 1F 04 00 1B
7998: C1 08 30 36 F6 07 20 00 EB
79A0: 04 E0 09 1F 36 36 6E 24 6C
79A8: 00 11 3E 27 24 24 04 00 D3
79B0: 21 24 17 07 30 36 36 0D 64
79B8: 24 00 24 1C 36 36 6E 3C A7
79C0: 0C 24 24 00 21 24 3F 36 95
79C8: 36 2E 25 04 00 28 E0 37 E0
79D0: 36 36 04 00 21 3C F0 36 FE
79D8: 36 0E 3D 28 24 00 05 20 DF
79E0: 1C 37 36 36 0D 24 00 38 0B
79E8: 60 B5 32 1E 27 00 12 24 23
79F0: 24 3C 0D 06 00 31 3E 27 6F
79F8: 24 24 0D 36 06 00 21 24 A2
7A00: 1F 36 36 0E 05 20 00 21 83
7A08: 24 1F 36 36 2E 28 26 00 5F
7A10: 2C 20 1F B6 32 0D 24 00 80
7A18: 12 24 64 FC 36 00 2C 20 A5
7A20: 3F 96 32 2D 04 00 32 00 F7
7A28: 00 00 00 00 00 00 00 00 1D
7A30: 00 00 00 00 00 00 00 00 25
7A38: 00 00 00 00 00 00 00 00 2D
7A40: 00 00 00 00 00 00 00 00 35
7A48: 00 00 00 00 00 00 00 00 3D
7A50: 00 00 00 00 00 00 00 00 45
7A58: 00 00 00 00 00 00 00 00 4D
7A60: 00 00 00 00 00 00 00 00 55
7A68: 00 00 00 00 00 00 00 00 5D
7A70: 00 00 00 00 00 00 00 00 65
7A78: 00 00 00 00 00 00 00 00 6D
7A80: 00 00 00 00 00 00 00 00 75
7A88: 00 00 00 00 00 00 00 00 7D
7A90: 00 00 00 00 00 00 00 00 85
7A98: 00 00 00 00 00 00 00 00 8D
7AA0: 00 00 00 00 00 00 00 00 95
7AA8: 00 00 00 00 00 00 00 00 9D
7AB0: 00 00 00 00 00 00 00 00 A5
7AB8: 00 00 00 00 00 00 00 00 AD
7AC0: 00 00 00 00 00 00 00 00 B5
7AC8: 00 00 00 00 00 00 00 00 BD
7AD0: 00 00 00 00 00 00 00 00 C5
7AD8: 00 00 00 00 00 00 00 00 CD
7AE0: 00 00 00 00 00 00 00 00 D5
7AE8: 00 00 00 00 00 00 00 00 DD
7AF0: 00 00 00 00 00 00 00 00 E5
7AF8: 00 00 00 00 00 00 00 00 ED
7B00: 00 00 00 00 00 00 00 00 F6
7B08: 00 00 00 00 00 00 00 00 FE
7B10: 00 00 00 00 00 00 00 00 07
7B18: 00 00 00 00 00 00 00 00 0F
7B20: 00 00 00 00 00 00 00 00 17
7B28: 00 00 00 00 00 00 00 00 1F
7B30: 00 00 00 00 00 00 00 00 27
7B38: 00 00 00 00 00 00 00 00 2F
7B40: 00 00 00 00 00 00 00 00 37
7B48: 00 00 00 00 00 00 00 00 3F
7B50: 00 00 00 00 00 00 00 00 47
7B58: 00 00 00 00 00 00 00 00 4F
7B60: 00 00 00 00 00 00 00 00 57
7B68: 00 00 00 00 00 00 00 00 5F
7B70: 00 00 00 00 00 00 00 00 67
7B78: 00 00 00 00 00 00 00 00 6F
7B80: 00 00 00 00 00 00 00 00 77
7B88: 00 00 00 00 00 00 00 00 7F
7B90: 00 00 00 00 00 00 00 00 87
7B98: 00 00 00 00 00 00 00 00 8F
7BA0: 00 00 00 00 00 00 00 00 97
7BA8: 00 00 00 00 00 00 00 00 9F
7BB0: 00 00 00 00 00 00 00 00 A7
7BB8: 00 00 00 00 00 00 00 00 AF
7BC0: 00 00 00 00 00 00 00 00 B7
7BC8: 00 00 00 00 00 00 00 00 BF
7BD0: 00 00 00 00 00 00 00 00 C7
7BD8: 00 00 00 00 00 00 00 00 CF
7BE0: 00 00 00 00 00 00 00 00 D7
7BE8: 00 00 00 00 00 00 00 00 DF
7BF0: 00 00 00 00 00 00 00 00 E7
7BF8: 00 00 00 00 00 00 00 00 EF
```

Program 4: SHAPETABLE5X7

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

```
7800: 64 00 CA 00 D1 00 D9 00 BE
7808: E7 00 F5 00 03 01 10 01 E8
7810: 17 01 1E 01 24 01 31 01 29
7818: 39 01 3E 01 44 01 48 01 75
7820: 4E 01 5C 01 64 01 71 01 1F
7828: 7C 01 87 01 92 01 9F 01 71
7830: A9 01 B7 01 C2 01 C9 01 EB
7838: D2 01 D9 01 E2 01 E9 01 0E
7840: F3 01 03 02 0F 02 1E 02 AA
7848: 28 02 35 02 41 02 4A 02 3D
7850: 56 02 62 02 6A 02 72 02 9B
7858: 7E 02 86 02 92 02 9E 02 D5
7860: A9 02 B3 02 BE 02 CA 02 D2
7868: D7 02 DF 02 EA 02 F6 02 31
7870: 02 03 0D 03 17 03 23 03 03
7878: 2A 03 32 03 38 03 40 03 07
7880: 45 03 4B 03 58 03 63 03 07
7888: 6C 03 76 03 81 03 8A 03 9F
7890: 97 03 9F 03 A6 03 AD 03 D1
7898: B7 03 BE 03 C7 03 CF 03 1B
78A0: D8 03 E3 03 ED 03 F4 03 D3
78A8: FD 03 07 04 0F 04 18 04 37
78B0: 23 04 2D 04 37 04 41 04 6A
78B8: 49 04 4F 04 57 04 61 04 0B
78C0: 73 04 7C 04 86 04 90 04 A5
78C8: 99 04 09 04 18 08 24 24 36
78D0: 00 01 40 18 20 6C 36 04 AE
78D8: 00 21 3C 0C 3C 0C 6C 36 7B
78E0: 6F 17 3E 0D 17 26 00 09 42
78E8: 1C 2D 0C 1C 37 1C 07 28 D6
78F0: 28 E5 16 04 00 2C 28 28 9B
78F8: 28 20 DF 33 25 8D 92 32 EB
7900: 25 04 00 29 0D 1C DF 6C 1E
7908: 29 DC 2F 38 60 15 04 00 8F
7910: 40 18 29 20 3C 26 00 49 14
7918: 1C 24 24 0C 04 00 29 20 F9
7920: 24 E4 04 00 01 28 28 C8 A0
7928: 23 1C 0D 16 B5 23 D8 27 C9
7930: 00 09 20 24 FC 12 6D 25 DB
7938: 00 21 35 F6 04 00 40 18 42
7940: 2D 2D 04 00 21 35 04 00 7B
7948: 28 28 28 28 20 00 20 24 46
7950: 64 2D 15 36 36 1E 3F 04 73
7958: 28 28 20 00 29 3D 20 24 10
7960: 24 1E 04 00 2D 2D DC 1B 60
7968: 64 2D 05 20 1C 3F 1E 04 99
7970: 00 70 2D 05 20 1C 2F 28 6D
7978: E0 3F 04 00 49 24 FD 3F 42
7980: 2C 28 28 30 36 04 00 70 CD
7988: 2D 05 20 E4 3F 27 2C 2D C1
7990: 25 00 29 2D 20 1C 3F 37 35
7998: 26 18 08 64 2D 04 00 09 6E
79A0: 24 05 20 05 20 3F 3F 04 BB
79A8: 00 29 2D 20 1C 3F 17 26 BF
79B0: 18 08 64 2D 15 26 00 29 7B
79B8: 2D 20 24 E4 3F 17 76 2D 8D
79C0: 04 00 29 3C 04 28 3C 04 DB
79C8: 00 91 05 20 27 25 38 2C 2D
79D0: 04 00 48 39 38 60 0C 04 C1
79D8: 00 18 08 2D 2D 04 38 3F CE
79E0: 27 00 08 0C 0C 1C 1C 04 36
79E8: 00 09 04 28 28 20 07 38 29
79F0: 1E 04 00 29 2D 18 08 24 84
79F8: 3F 36 FD 1A 24 24 0C 2D 71
7A00: 15 04 00 24 24 0C 0C 15 41
7A08: 15 36 36 C4 3F 04 00 2D 5F
7A10: 2D 20 1C 3F 36 04 18 08 15
7A18: 3C 28 2D 15 26 00 20 24 C1
7A20: 64 2D 15 96 F2 3F 04 00 3B
7A28: 05 20 24 3C 28 2D 15 36 46
7A30: 36 1E 3F 04 00 24 24 24 EC
7A38: 2D 2D 96 3B B7 2A 2D 04 5A
7A40: 00 24 24 24 2D 2D 96 3B 8B
7A48: 27 00 20 24 64 2D 15 9E B7
7A50: 2B 35 F6 3F 04 00 24 24 87
7A58: 24 4D 31 36 3F 6F 11 36 4C
7A60: 04 00 29 E5 24 24 07 28 C2
7A68: 25 00 A8 2D 20 24 3C 28 0A
7A70: 25 00 24 24 24 4D F1 1E 17
7A78: 1E 0E 0E 0E 04 00 2D 2D 4A
7A80: DC 1B 24 24 24 00 24 24 FE
7A88: 24 0E 15 66 28 30 36 36 C0
7A90: 26 00 24 24 24 0E 0E 0E E2
7A98: 96 21 24 24 24 00 20 24 6D
7AA0: 64 2D 15 36 36 1E 3F 04 C5
7AA8: 00 24 24 24 2D AD 36 1E 18
7AB0: 3F 04 00 20 24 64 2D 15 6A
7AB8: 36 B6 1F 67 21 00 24 24 46
7AC0: 24 2D AD F6 3F 0E 0E 0E 94
7AC8: 04 00 A8 2D 05 20 1C 3F C7
7AD0: 07 20 0C 2D 15 04 00 09 67
7AD8: 24 24 24 3F 4D 25 00 20 80
7AE0: 24 24 4D 31 36 36 1E 3F B3
7AE8: 04 00 09 07 38 20 24 6C 68
7AF0: 09 36 36 1E 04 00 20 24 25
7AF8: 24 4D 31 36 36 1E 1F 0C 51
7B00: 24 00 64 0C 0C 0C FC 1B FB
7B08: 76 8E 71 26 00 09 24 3C 17
7B10: 38 20 4D 31 1E 04 00 2D 16
7B18: 2D DC 1B 0C 0C 0C 0C 3C E5
7B20: 3F 27 00 09 E5 24 24 2C 45
7B28: 04 00 48 09 1C 1C 1C 1C 60
7B30: 04 00 29 24 24 24 27 00 90
7B38: 40 18 28 28 70 0E 04 00 A0
7B40: 12 2D 2D 04 00 40 40 18 0B
7B48: 70 04 00 29 0D 07 20 3F 0F
7B50: 17 05 40 18 2D 15 04 00 63
7B58: 29 2D 20 E4 3F 36 26 40 E0
7B60: 18 24 00 29 E5 1B 24 0C EE
7B68: 2D 15 04 00 20 64 2D 1D C5
7B70: 20 96 36 3E 27 00 29 25 78
7B78: 08 3F 3F 26 60 2D 15 04 73
7B80: 00 21 24 3C 05 60 2D D6 E2
7B88: 27 00 29 2D 36 3F 27 18 20
7B90: 08 24 2C 2D 36 26 00 24 5B
7B98: 24 24 95 AD 36 26 00 29 AB
7BA0: E5 24 3C 0C 20 00 72 2D EE
7BA8: 20 24 24 27 00 24 24 24 AC
7BB0: 4D F2 1E 0E 0E 26 00 29 E1
```

```
7BB8: 3D 20 24 24 27 00 24 24 C2
7BC0: AD 36 6E 24 24 27 00 21 0B
7BC8: 24 1C 0D AD 36 26 00 20 BF
7BD0: 64 2D 15 36 1E 3F 04 00 41
7BD8: 12 24 24 24 2D AD 36 1E 53
7BE0: 3F 04 00 20 64 2D 15 36 B2
7BE8: 36 E6 38 27 00 21 24 1C 17
7BF0: 0D AD 04 00 29 2D E0 3F 59
7BF8: 07 60 2D 04 00 71 05 E0 22
7C00: 23 24 67 AC 2A 04 00 20 CC
7C08: 24 4D 36 76 1F 27 00 09 33
7C10: 0C 0C 24 DF 33 76 04 00 10
7C18: 29 20 56 05 20 24 DF 33 4D
7C20: 36 04 00 0C 0C 1C 1C 4D 4C
7C28: F1 16 0E 04 00 12 2D 20 64
7C30: 24 1C 1C 4D F1 04 00 2D 67
7C38: 2D DC 63 0C 0C 3F 3F 04 0C
7C40: 00 09 E5 23 1C 0C 64 25 69
7C48: 00 09 24 24 24 04 00 29 A4
7C50: 05 20 0C 1C E4 27 00 40 1B
7C58: 18 18 08 0C 15 15 05 20 4C
7C60: 00 29 2D 38 3F 67 2D 05 C3
7C68: 38 3F 67 2D 05 38 3F 67 FB
7C70: 2D 04 00 09 24 24 F4 1E 4B
7C78: 4D E1 04 00 18 08 0E 0E 1C
7C80: 2C 0C 1F 24 24 00 09 E0 CC
7C88: 1C 05 30 25 A8 2A 04 00 1F
7C90: 09 28 28 38 3C B0 3F 04 C7
7C98: 00 25 24 2C 1F 05 60 AD 9D
7CA0: 96 F2 07 20 00 20 00 00 05
```

Program 5: SHAPETABLE7X9

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

```
7800: 80 00 02 01 10 01 20 01 47
7808: 2E 01 3B 01 46 01 54 01 A7
7810: 61 01 73 01 7A 01 87 01 58
7818: 93 01 9E 01 A8 01 B5 01 AC
7820: C0 01 CD 01 DA 01 E7 01 27
7828: F1 01 FC 01 08 02 14 02 74
7830: 24 02 33 02 40 02 4C 02 DE
7838: 57 02 63 02 6E 02 7A 02 54
7840: 8B 02 9C 02 A4 02 AD 02 B5
7848: C2 02 DB 02 ED 02 00 03 31
7850: 07 03 10 03 19 03 33 03 F5
7858: 3E 03 44 03 4B 03 4E 03 E7
7860: 56 03 6A 03 74 03 84 03 76
7868: 95 03 A5 03 B4 03 C7 03 0E
7870: D3 03 E7 03 F8 03 00 04 11
7878: 08 04 13 04 1E 04 29 04 68
7880: 35 04 48 04 59 04 6C 04 0E
7888: 7A 04 89 04 99 04 A4 04 53
7890: B4 04 C5 04 D0 04 DA 04 26
7898: EA 04 F5 04 06 05 18 05 78
78A0: 28 05 35 05 47 05 58 05 E1
78A8: 68 05 73 05 81 05 90 05 14
78B0: A3 05 B3 05 C0 05 D0 05 3C
78B8: DB 05 E5 05 EE 05 F8 05 68
78C0: FD 05 04 06 10 06 20 06 B1
78C8: 2B 06 3B 06 48 06 53 06 9F
78D0: 65 06 71 06 7A 06 84 06 7F
78D8: 91 06 98 06 A4 06 AE 06 28
78E0: B9 06 C8 06 D9 06 E1 06 5A
78E8: EC 06 F7 06 02 07 0C 07 7C
78F0: 19 07 25 07 36 07 43 07 A0
78F8: 4E 07 56 07 60 07 6A 07 09
7900: 8B 07 20 64 AD 15 15 15 C5
7908: 04 40 18 1E 9F F2 27 00 09
7910: 92 05 20 24 24 24 2D AD 8D
7918: 1E 3F 96 2D 05 20 04 00 40
7920: 92 29 3C 2C 24 24 1C 6F 4A
7928: 49 F1 1E 1E 04 00 29 05 59
7930: 20 D7 23 0C E5 07 20 64 3E
7938: AD 04 00 09 E5 3B 20 2D 1D
7940: E5 1B 0C 0C 25 00 09 2D 97
7948: E0 3F 07 20 05 28 20 74 DB
7950: 2D 3C 04 00 92 49 09 24 59
7958: 24 24 1C FF 17 0D 36 26 69
7960: 00 09 2D 28 20 3C 3F 3F 6D
7968: 36 0E C4 18 08 64 0C AD AB
7970: 15 26 00 29 05 F8 23 24 80
7978: 24 00 24 24 24 95 2A 28 38
7980: F0 12 0E 0E 05 20 00 05 C0
7988: 28 28 15 15 C4 D8 24 E4 44
7990: 1C 04 00 92 24 24 24 6C 22
7998: 31 36 0E 1F 27 00 21 24 05
79A0: 24 6F 49 F6 1E 1E 04 00 8B
79A8: 49 2D E0 3F 3F 60 0C E5 15
79B0: 63 2D 1C 04 00 09 2D 28 0B
79B8: 20 1C 1C F7 1E 76 04 00 98
79C0: 09 24 24 3C 17 4D 28 2D F4
79C8: 9F 36 36 04 00 92 24 24 D6
79D0: 24 0C 0C AD 15 F6 1E E7 DD
79D8: 04 00 18 08 64 0C 2D 2D 2C
79E0: 9F 0E F6 1E 3F 20 00 49 AB
79E8: 24 24 3C BF 0D 48 2D 04 62
79F0: 00 01 20 24 3C 4D 09 17 AA
79F8: 36 F6 27 00 09 24 3F 20 21
7A00: 64 25 B4 A9 36 1E 04 00 D3
7A08: 29 28 28 28 F8 1B 6F AA E0
7A10: 11 15 25 00 92 49 24 24 9D
7A18: 24 24 DF 0E 36 0E 0D 05 0E
7A20: 20 64 04 00 29 05 20 24 80
7A28: D8 17 36 6E 4A 2D 20 24 68
7A30: 1C 04 00 25 E4 24 0C 0C 62
7A38: 2D 15 0E 36 17 36 25 00 0A
7A40: 18 08 0E 0E 24 24 24 04
7A48: 2D 2D 04 00 40 18 28 2D 7F
7A50: 2D E5 1C 96 2A 20 00 40 54
7A58: 18 28 28 B0 12 07 20 2D 8D
7A60: 2D 25 00 49 24 24 24 E7
7A68: 1E 1E 4D E1 04 00 49 18 86
7A70: 08 04 38 3F 44 09 56 2D 85
7A78: 04 00 2D 2D 2D DC DB 0C 88
7A80: 0C 0C FD 38 38 38 28 2D E1
7A88: 2D 25 00 40 18 05 28 15 9B
7A90: 15 2D 20 18 08 1E 3F 38 50
7A98: 38 17 04 00 49 04 18 08 82
7AA0: 24 24 24 00 40 40 18 08 70
7AA8: 24 4D 36 04 00 09 24 3C B2
7AB0: 27 28 35 0E 35 36 44 28 FD
7AB8: 04 38 37 1C 1C 24 0D 36 28
7AC0: 04 00 28 2D 2D 05 20 1C 69
7AC8: 3F 3F 07 20 0C 2D 2D E5 65
7AD0: B3 B6 B6 1F 04 20 04 20 DE
7AD8: 04 20 00 49 09 05 E0 17 A1
7AE0: DF 63 0C 0C 0C 0C 0C DF 69
7AE8: E3 17 0E 04 00 29 6D E1 F8
7AF0: 27 65 DF 1F 17 2E 40 60 13
7AF8: 1F 07 20 0C 2D 15 26 00 0E
7B00: 40 C0 18 08 0C 24 00 49 05
7B08: 1C 1C 24 24 0C 0C 04 00 73
7B10: 49 0C 0C 24 24 1C 1C 04 40
7B18: 00 60 0C 2D 0E 0E DE 23 05
7B20: 24 38 3F 04 18 08 0E 0E 6A
7B28: 2D 0C 0C 96 3F 07 18 08 F1
7B30: 24 04 00 49 20 24 3F 67 46
7B38: 09 24 95 2A 25 00 92 29 89
7B40: 20 3C 04 00 40 18 28 2D B6
7B48: 2D 25 00 09 04 00 60 0C 9C
7B50: 0C 0C 0C 0C 04 00 29 2D 32
7B58: 2D 20 24 24 E4 3F 3F 17 6E
7B60: 36 36 36 0C 0C 0C 0C 0C 3C
7B68: 04 00 29 2D E5 23 24 24 81
7B70: 24 1E 04 00 2D 2D 2D DC D6
7B78: DB 63 0C 0C 65 0C E4 3F DC
7B80: 3F 17 04 00 70 2D 2D 05 F4
7B88: 20 E4 3F 27 18 18 0C 2D 89
7B90: 2D 15 36 04 00 49 09 24 C5
7B98: 2C 1F 3F 3F 0C 0C 0C 0C FD
7BA0: 0C 36 36 04 00 70 2D 2D 7B
7BA8: 05 20 E4 3F 3F 27 24 2C C5
7BB0: 2D 2D 25 00 29 2D 2D 20 A6
7BB8: E4 3F 3F 37 36 04 40 18 A7
7BC0: 24 0C 2D 2D 15 04 00 09 07
7BC8: 24 64 0C 0C 0C 3C 3F 3F 3C
7BD0: 37 04 00 29 2D 2D 20 E4 3A
7BD8: 3F 3F 17 36 04 40 18 24 FA
7BE0: 0C 2D 2D 15 36 04 00 70 52
7BE8: 2D 2D 05 20 24 3F 3F 3F 40
7BF0: 20 64 2D 2D 15 36 04 00 13
7BF8: 01 18 08 04 40 18 04 00 22
7C00: 92 61 24 27 40 18 20 00 34
7C08: 49 E1 1C 1C 1C 0C 0C 0C 98
7C10: 0C 04 00 40 18 2D 2D 2D 11
7C18: 04 38 3F 3F 27 00 09 05 4D
7C20: 28 28 28 E0 1C 1C 1C 04 D7
7C28: 00 49 04 20 0C 0C 0C E4 83
7C30: 3F 3F 17 26 00 29 2D E5 C2
7C38: DB 23 24 24 64 2D 2D 15 F5
7C40: 36 F6 3F 27 0C 35 04 00 A9
7C48: 24 24 24 0C 0C 2D 15 15 F5
7C50: 36 3F 3F 6F 4A 31 36 04 9A
7C58: 00 2D 2D 2D 20 E4 3F 3F 67
7C60: 36 26 18 40 24 3C 0D 2D 5E
7C68: AD 36 04 00 09 2D 2D F8 96
7C70: DB 1C 24 24 0C 0C 2D AD BD
7C78: 04 00 2D 2D 2D 20 24 24 42
7C80: E4 3F 3F AF 36 36 36 04 99
7C88: 00 2D 2D 2D DC DB 23 24 06
7C90: 2D E5 1B 24 2C 2D 2D 25 D4
7C98: 00 24 24 2D E5 1B 24 2C 02
7CA0: 2D 2D 25 00 09 2D 2D 20 97
7CA8: 3C FF 9A 1C 24 24 0C 0C AA
7CB0: 2D AD 04 00 24 24 24 24 4A
7CB8: 4D 49 36 36 36 36 C4 18 01
7CC0: 08 3F 3F 04 00 29 2D E5 9A
7CC8: 23 24 24 24 3F 4D 25 00 9C
7CD0: 70 2D 05 20 24 24 24 6F 59
7CD8: 04 00 24 24 24 24 4D 49 30
7CE0: 1E 1E 1E 1E 6F 72 0E 0E 85
7CE8: 04 00 2D 2D 2D DC DB 23 14
7CF0: 24 24 24 04 00 24 24 24 C6
7CF8: 24 0E 0E 0E 26 08 0C 0C 9F
7D00: 36 36 36 36 04 00 24 24 5A
7D08: 24 24 0E 0E 0E 0E 0E 0E 93
7D10: 36 04 18 08 24 24 24 00 A4
7D18: 09 2D 05 28 20 24 E4 1C 7D
7D20: 3F 17 17 36 36 0E 04 00 B8
7D28: 24 24 24 24 2D 2D AD 36 B4
7D30: 1E 3F 3F 04 00 09 2D 0D BD
7D38: 1C 1C 0D 24 24 1C 1C 3F 35
7D40: 1E 1E 36 36 0E 04 00 24 A0
7D48: 24 24 24 2D 2D AD 36 1E 60
7D50: 3F 3F 8D 0E 0E 0E 04 00 FD
7D58: 70 2D 2D 05 20 E4 3F 3F 1F
7D60: 07 20 64 2D 2D 15 04 00 0C
7D68: 49 24 24 24 24 3F 6F 09 DD
7D70: 2D 04 00 29 2D 2D 20 24 18
7D78: 24 24 DF DB 36 36 36 26 65
7D80: 00 49 1C 1C 1C 24 24 6C 39
7D88: 49 31 36 36 1E 1E 04 00 10
7D90: 24 24 24 24 4D 49 36 36 9F
7D98: 36 36 07 38 38 20 B4 1A 66
7DA0: 1E 04 00 64 0C 0C 1C 1C D6
7DA8: 1C 6C 49 31 1E 1E 16 0E AC
7DB0: 0E 26 00 49 24 24 1C 1C D6
7DB8: 1C 6C 49 31 1E 1E 04 00 8A
7DC0: 2D 2D 2D DC DB 63 0C 0C A1
7DC8: 0C 0C 0C 3C 3F 3F 27 00 57
7DD0: 09 2D E5 1B 24 24 24 2C 30
7DD8: 2D 04 00 48 49 E1 1C 1C 16
7DE0: 1C 1C 1C 04 00 29 2D 24 D7
7DE8: 24 24 24 3F 27 00 40 40 71
7DF0: 40 48 E1 3F 3F 17 04 00 AC
7DF8: 2D 2D 2D 04 00 40 40 40 7D
7E00: 49 1C 24 00 29 6D 3C 05 A9
7E08: 20 24 3F BF 17 36 04 00 9B
7E10: 24 24 65 2D 15 36 F6 3F 56
7E18: 07 20 18 18 08 24 24 00 3E
7E20: 29 2D 05 F8 DB 24 64 2D 92
7E28: AD 04 00 29 6D 3C 05 20 16
7E30: 3C 05 20 24 96 1B 3F 17 89
7E38: 36 26 00 29 2D E5 DB 23 48
7E40: 2D 2D 25 E4 3F BF 26 00 57
7E48: 09 24 24 3F 4D FD 20 64 52
7E50: AD 04 00 12 0E 2D 2D 20 E5
7E58: 24 24 24 1E 07 38 BF 36 A5
7E60: 76 2D 05 20 00 24 24 24 83
7E68: 24 96 0A 0C AD 15 36 26 73
7E70: 00 49 24 24 3C 05 18 08 B4
7E78: 04 00 12 0E 2D 2D 20 24 1D
7E80: 24 24 04 00 24 24 24 24 37
7E88: 96 49 F1 1E 6F 72 0E 04 A8
7E90: 00 09 24 24 24 24 04 00 50
7E98: 24 24 2C AD 36 36 4D 24 5A
7EA0: 24 1C 27 00 24 24 AC 0C B2
7EA8: 2D 15 36 36 04 00 29 2D 4B
7EB0: 05 20 24 1C 3F BF 36 26 0A
7EB8: 00 92 24 24 24 24 0E 05 F3
7EC0: 28 AD 36 F6 3F 07 20 00 C9
7EC8: 92 49 09 24 24 24 24 1E DC
7ED0: 07 38 BF 36 76 2D 05 20 4D
7ED8: 00 24 24 74 0C 2D 15 04 ED
7EE0: 00 70 2D 2D E0 E7 E7 0C F4
7EE8: 2D AD 04 00 49 2D F8 23 7C
```

```
7EF0: 24 24 B4 3B 4D 25 00 29 7B
7EF8: 2D 05 20 24 FC DB 36 36 0E
7F00: 04 00 09 1C 1C 24 6C 09 37
7F08: 36 F6 04 00 20 24 6C 11 DB
7F10: 36 F6 6F 29 05 20 24 24 7D
7F18: 00 0C 0C 25 3F 38 68 49 E2
7F20: 1E 96 15 04 00 12 0E 2D 48
7F28: 2D 20 24 24 24 DF 1B 36 99
7F30: 36 0E 2D 05 20 00 2D 2D 4C
7F38: E5 DB 0C 0C 0C 05 38 3F 87
7F40: 3F 04 00 49 2D DC 23 E4 7C
7F48: 0C 24 0C 2D 04 00 49 20 7D
7F50: 24 18 08 24 04 00 29 2D 4A
7F58: 20 64 1C 24 1C 3F 04 00 2C
7F60: 40 18 40 18 60 AD 15 2D 20
7F68: 20 00 24 24 24 24 35 36 90
7F70: 36 36 2E 24 24 24 24 35 4F
7F78: 36 36 36 2E 24 24 24 24 E7
7F80: 35 36 36 36 2E 24 24 24 40
7F88: 24 04 00 00 00 FF 00 00 9A
```

Program 6: DISPLAYSHAPE

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

```
2E 10 HOME : TEXT
92 20 HGR : SCALE= 1: ROT= 0: HC
      OLOR= 3
A0 30 POKE 232,0: POKE 233,112
F8 40 VTAB 22: PRINT "ENTER NAME
       OF TABLE:": INPUT "";N$:
      PRINT "ENTER DRIVE #: ";:
      GET AN$:AN = VAL (AN$)
4C 50 PRINT AN
9E 60 PRINT CHR$ (4);"BLOAD ";N$
      ;",A28672,D";AN
E3 70 X = 1:Y = 1
3D 80 FOR I = 0 TO 200 STEP 20
48 90 FOR J = 1 TO 20
23 100 X = X + 10: IF J = 1 THEN
       X = 1:Y = Y + 15
F9 110 IF I + J > PEEK (28672) T
      HEN 150
C8 120 DRAW I + J AT X,Y
64 130 NEXT J
E5 140 NEXT I
1A 150 HOME : VTAB 22: PRINT "PL
      EASE PRESS A KEY ";: GET
      AN$: TEXT : HOME : END
```


Program 7: BARCHART

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

```
B7 110 TEXT : HOME :MAX = 0
2F 120 READ NB,WB
3B 130 WS = (280 - NB * WB) / (N
      B + 1)
F7 140 X = WS + 1
74 150 IF NB * (WB + WS) < 280 T
      HEN 170
5B 160 PRINT CHR$ (7);"CHART IS
      TOO WIDE": PRINT "PLEASE
      PRESS A KEY ": GET AN$: G
      OTO 490
48 170 DIM BAR(2 * NB),C(50)
2A 180 FOR I = 1 TO 2 * NB
98 190 READ BAR(I)
99 200 IF MAX < BAR(I) THEN MAX
      = BAR(I)
E0 210 NEXT I
11 220 SCL = 130 / MAX
5B 230 HGR : ROT= 0: SCALE= 1: H
      COLOR= 3
0C 240 HPLOT 0,0 TO 0,159 TO 279
      ,159 TO 279,0 TO 0,0
FC 250 FOR I = 1 TO NB
7D 260 HPLOT X,159 TO X,159 - SC
      L * BAR(I) TO X + WB,159
      - SCL * BAR(I) TO X + WB,
      159
91 270 X = X + WB + WS
EE 280 NEXT I
5D 281 X = WS + WB + 13:BW = WB
      - 5
FB 282 FOR I = NB + 1 TO 2 * NB
4B 283 HPLOT X,159 TO X,159 - SC
      L * BAR(I) TO X + BW,159
      - SCL * BAR(I) TO X + BW,
      159
A3 284 X = X + WB + WS
03 285 NEXT I
FF 290 POKE 232,0: POKE 233,112
19 300 PRINT CHR$ (4);"BLOAD BAR
      TABLE,A$7000"
F5 310 FOR I = 1 TO NB
2C 320 K = 0
30 330 FOR J = 1 TO 10
55 340 READ C(J): IF C(J) = 0 TH
      EN 380
4A 350 DRAW C(J) AT I * (WS + WB
      ) + 5,156 - K * 9
5B 360 K = K + 1
6E 370 NEXT J
78 380 IF I = NB THEN 400
F1 390 NEXT I
43 400 FOR I = 1 TO 25
FB 410 READ C(I)
E3 420 DRAW C(I) AT 18 + (I * 9)
      ,13
E6 430 NEXT I
FF 440 FOR I = 1 TO 4
04 450 READ C(I)
07 460 DRAW C(I) AT 185 + (I * 5
      ),22
EE 470 NEXT I
0A 471 FOR I = 1 TO 4
DB 473 READ C(I): DRAW C(I) AT 5
      7 + I * 5,22
FE 474 NEXT I
6D 476 DRAW 32 AT 85,24: DRAW 31
       AT 213,24
F1 480 DRAW 30 AT 10,150
8E 490 VTAB 24: PRINT "PRESS ANY
       KEY ";: GET AN$: TEXT :
      HOME
8D 500 END
AC 510 DATA 4,10,160,108,42,168,
      110,148,111,127
F5 520 DATA 1,2,3,0,4,2,2,3,0,7,
      1,6,1,5,0,8,5,9,4,3,8,0
72 530 DATA 10,11,12,13,14,15,16
      ,11,24,17,14,11,11,12,24,
      18,10,19,20,21,22,14,15,1
      9,23
23 540 DATA  25,26,27,28,25,26,2
      9,25
```

Program 8: BARTABLE

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

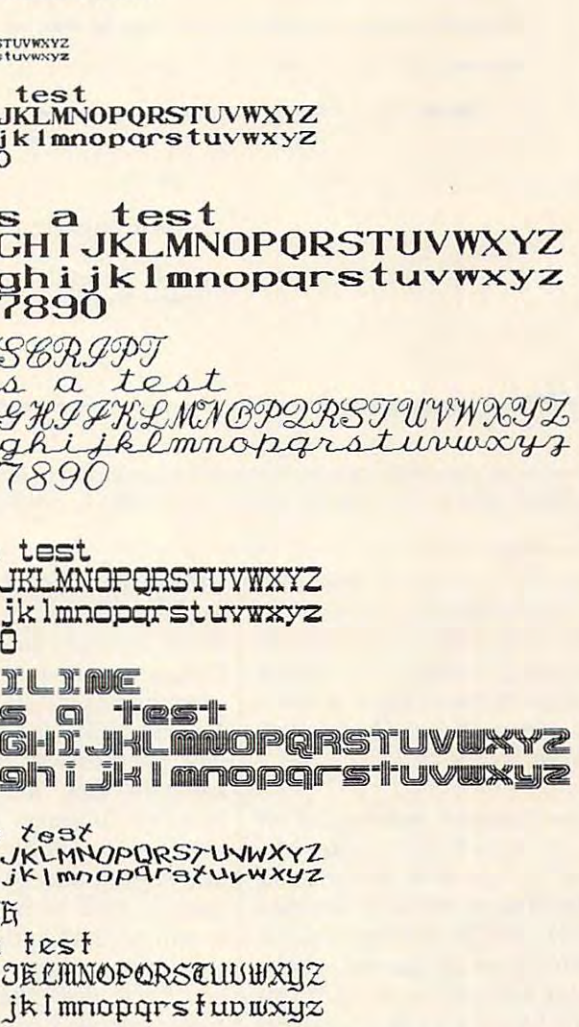
```
7800: 20 00 42 00 4E 00 5B 00 72
7808: 66 00 72 00 7A 00 84 00 57
7810: 90 00 9C 00 A7 00 B8 00 8B
7818: C8 00 D3 00 E4 00 EF 00 EE
7820: FA 00 09 01 19 01 26 01 D9
7828: 36 01 45 01 53 01 61 01 8F
7830: 73 01 75 01 7C 01 83 01 C9
7838: 8B 01 92 01 9C 01 D5 01 27
7840: E4 01 24 24 0C 0C 15 15 7A
7848: 36 36 C4 3F 04 00 A8 2D 0D
7850: 05 20 1C 3F 07 20 0C 2D 41
7858: 15 04 00 20 24 24 4D 31 54
7860: 36 36 1E 3F 04 00 24 24 3E
7868: 24 2D AD F6 3F 0E 0E 0E 38
7870: 04 00 A8 2D 20 24 3C 28 7D
7878: 25 00 24 24 24 2D AD 36 2A
7880: 1E 3F 04 00 24 24 24 0E D8
7888: 0E 0E 96 21 24 24 24 00 E2
7890: 24 24 24 2D 2D 96 3B B7 E5
7898: 2A 2D 04 00 20 24 64 2D F1
78A0: 15 36 36 1E 3F 04 00 24 80
78A8: 24 24 24 2D 2D AD 36 1E B6
78B0: 3F 3F 8D 0E 0E 0E 04 00 54
78B8: 2D 2D 2D DC DB 23 24 2D DF
78C0: E5 1B 24 2C 2D 2D 25 00 1B
78C8: 2D 2D 2D DC DB 23 24 24 E6
78D0: 24 04 00 24 24 24 0C 0C EC
78D8: 2D 15 15 36 3F 3F 6F 4A CB
78E0: 31 36 04 00 49 24 24 24 BF
78E8: 24 3F 6F 09 2D 04 00 29 DC
78F0: 2D E5 23 24 24 24 3F 4D 16
78F8: 25 00 49 1C 1C 1C 24 24 25
7900: 6C 49 31 36 36 1E 1E 04 6F
7908: 00 70 2D 2D 05 20 E4 3F 41
7910: 3F 07 20 64 2D 2D 15 04 FA
7918: 00 24 24 24 24 2D 2D AD B8
7920: 36 1E 3F 3F 04 00 09 2D F0
7928: 05 28 20 24 E4 1C 3F 17 1B
7930: 17 36 36 0E 04 00 2D 2D 8B
7938: 2D 20 24 24 E4 3F 3F AF E2
7940: 36 36 36 04 00 29 2D 2D 0F
7948: 20 24 24 24 DF DB 36 36 2C
7950: 36 26 00 09 2D 2D F8 DB 64
7958: 1C 24 24 0C 0C 2D AD 04 1C
7960: 00 24 24 24 24 0E 0E 0E A6
7968: 0E 0E 0E 36 04 18 08 24 BF
7970: 24 24 00 00 00 20 BC 95 0E
7978: 36 6F 04 00 38 24 2D 36 C5
7980: 36 26 00 38 64 95 32 1E 97
7988: 1C 24 00 38 24 2D 96 F6 10
7990: 27 00 38 28 70 D7 32 0E F5
7998: 05 20 04 00 24 24 24 24 B4
79A0: 2D 2D 2D 2D 2D 2D 2D 2D 93
79A8: 2D 2D 36 36 36 36 3F 3F EF
79B0: 3F 3F 3F 3F 3F 3F 3F 27 8B
79B8: 08 40 C0 0E 0E 0E 0E 0C B9
79C0: 0C 0C 0C 0E 0E 0E 0E 0C EF
79C8: 0C 0C 0C DF F3 9E F3 1C 62
79D0: 1C 18 1C 04 00 24 24 2C A0
79D8: 2D 2D 2D 2D 36 36 3E 3F 6C
79E0: 3F 3F 27 00 24 24 2C 2D 5F
79E8: 35 36 36 3F 3F 00 00 04 BC
```

Font Printer
For The IBM PC/PCjr

John Klein

"Font Printer" for the IBM PC/PCjr allows you to print a wide variety of custom character styles on a dot-matrix printer. Its editor makes it easy to design custom text fonts, and the printing program lets you print any ASCII (plain text) file using your custom print style. Another program allows you to print large banners using any custom printer font. As a special bonus, the quarterly IBM PC/PCjr disk that includes this month's COMPUTE! programs also contains a library of 25 ready-to-use custom printer fonts for this program. The editor program requires a color monitor, and, for the IBM PC and compatibles, a color/graphics card or equivalent hardware is also required. The printing program requires an IBM Graphics Printer or compatible dot-matrix printer. All the programs require BASICA for the PC, Cartridge BASIC for the PCjr, or GW-BASIC for compatibles, and DOS version 2.1 or higher.

Figure 1: Custom Fonts



"Font Printer" allows you to create many different custom fonts and print any font in a variety of sizes.

"Font Printer" makes it possible to create, edit, and print custom fonts on a dot-matrix printer. You can print text in almost any imaginable print style, from Gothic and Roman to Old English, outlined characters, or whatever else you can devise. You have full control over the shape of each character, so Font Printer isn't limited to printing ordinary characters of the alphabet. It also can print custom letterheads, other graphic designs, and banners.

This article includes four programs. The font editor (Program 1) lets you design and edit complete custom fonts and save them to disk. The printing program (Program 2) lets you print any ASCII text file using the custom font of your choice. Program 3 helps you keep track of the custom fonts you have created, and the banner printer program (Program 4) prints large-letter banners using custom fonts. The accompanying figures illustrate just a part of what you can do with Font Printer.

Figure 2: Letterhead

"Font Printer" can also create eye-catching graphic designs, such as this letterhead for a fictitious travel agency. The graphics shapes are created by redefining a block of text characters.

(Ed. Note: As a bonus for disk subscribers, the COMPUTE! Disk that includes the programs for this month also contains 25 ready-made custom printer fonts. (Because space is limited, we cannot publish the bonus fonts in the magazine.)

These programs were tested on an IBM Graphics Printer, a Star Micronics SD-10 (in IBM mode), and on an Okidata printer with the IBM Plug 'n Play Kit. To use Font Printer on another type of printer, check the printer manual to see if it can print APA (All Points Addressable) graphics, also known as bit-image graphics. The printer must also use the command sequence CHR$(27) CHR$(76)—ESC L—in order to get 960 dots in the same space normally occupied by 80 text characters—usually referred to as double-density graphics mode.

Using The Font Editor

Type in Programs 1–4 and save them on disk; then load and run Program 1, the font editor. The program begins by asking you to enter a filename for the font you wish to edit. Each custom font is saved in a separate disk file. The filename must be no longer than eight characters, not including the extension. You should add a special extension such as .FNT with font filenames as a reminder that they contain custom printer fonts.

Each disk that contains font files must also contain a font code directory file named FONTCODE.DIR. If no font code directory is found on the disk, the font editor will create a new FONTCODE.DIR file. The font editor will also make the appropriate entries in the font code directory for each new font you create. Any disks containing font files must also have a font code directory file before the fonts can be used by Programs 2–4.

To load an existing font, place a disk containing the appropriate font file in the drive and enter the desired filename. The editor loads the file from disk and displays the first character of that font on the screen. If you enter the name of a file that is not found on the disk or is not currently in the font code directory, Program 1 asks whether you wish to create a font code directory entry for the new file. Press Y to create a new font file or to add an existing file to the font code directory, or press N to abort the function and return to the original prompt.

If a file with the specified name exists on the disk but is not in the font code directory, Program 1 allows you to simply add the file to the directory and begin editing. Otherwise, you must create a new font file. In this case, you have two options: You can start with an empty font (all character patterns will be blank), or you can start with a font that is a copy of an existing font. If you choose the option to start with an empty font, you must answer a few questions before you begin to edit. First, the program requests that you specify the character size in terms of width and height.

The character width (in dots) can be any whole number from 4–36. A width value of 12 creates normal-width characters which will print ten characters to the inch on a standard printer. To create half-width characters, you would specify a width of 6. Use a width of 24 for double-width characters, 36 for triple-width, and so on.

After you specify the character width, Program 1 asks you to specify the character height in terms of lines. A character may be one, two, or three lines high. Each line contains eight vertical dots, and the normal printer font is one line in height. A character with a height of two lines is 16 dots high, and one with three lines is 24 dots high—a very large character, indeed.

The next prompt asks you to enter the *font call code* for this font. This is the code name you will use to call (begin using) the font within a word processing document. The

font call code can be a descriptive word of any length, but it must contain no spaces, colons, or hyphens. To use a font, you need to remember its call code, not its filename. The font code directory matches call codes with filenames.

Finally, the font editor program prompts you to press C if you wish to create a character font or H if you wish to create a header. A header is simply a picture or graphic design that is built of several individual custom characters. The process of creating headers is described later in this article.

Editing Screen

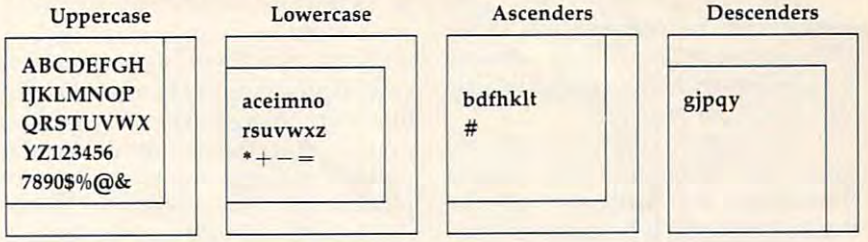
After you answer those questions (or if you began the program by loading an existing font), the font editor displays the main editing screen. This screen is divided into two areas. On the left is a list of single-key options used in editing. On the right is the editing window, which displays an enlarged version of the current character.

A custom character set is created by drawing one character at a time in the editing window. Use the cursor keys to move around inside the editing window. If you press the space bar when the cursor is on a blank space, that space is turned on. To turn off a space that is already turned on, simply move the cursor to that square and press the space bar again.

Font Printer allows you to define patterns for ASCII characters 33–126. These are all the characters that can be entered from the IBM PC/PCjr keyboard without using the Alt-key entry mode.

If you are creating a comparatively small font, you can draw each character by moving around the editing window and turning on the dots to represent that character. For larger characters or graphics, you may find it faster to draw each character on graph paper before transferring the design to the computer. Another method is to tape a sheet of clear plastic over the monitor screen and draw on the plastic with washable marking pens, then use the editor to fill in the squares to make up the design. When you're done creating one character, simply wipe off the plastic and proceed to the next.

Figure 3: Centering Guidelines



Placement

The placement of each character within the editing window is critical. If you don't align each character in the set properly, the font may look messy or be difficult to read. For most fonts, you'll want to leave white space around characters to prevent them from running into one another and becoming illegible. Figure 3 illustrates some centering guidelines which will create a pleasing appearance in most cases. Note that all characters except uppercase I should be placed flush against the left side of the editing window.

The exact amount of blank space bordering each character depends on the effect you wish to achieve. In general, you should leave one or two blank lines below the characters to leave room for descenders on the lowercase characters *g*, *j*, *p*, *q*, and *y*. Similarly, the top line or two of space should be reserved for uppercase characters and for the ascenders in the lowercase characters *b*, *d*, *h*, *k*, *l*, and *t*. Every character should have at least one row of white space to the right.

Editing Options

The left side of the editing screen displays the font editor's single-key commands:

M	move to new character
C	clear editing window
T	trace from character
S	save character to disk
I	reinitialize font parameters
A	clear all characters
W	rewrite screen
R	restart program
Q	quit
SPACE	plot/erase point or cancel current command
D	turn draw mode on/off
E	turn erase mode on/off

The move command (M) lets you change which character you are editing.

The clear window command (C) clears the editing window, erasing the current character pattern.

The trace command (T) allows you to copy the pattern of another character in the font into the editing window. This is useful for creating characters that look similar. For instance, to create a lowercase *e* character, you might begin by tracing in the pattern of lowercase *c* (assuming you have already created the *c*). When asked to choose which character to trace, you may either enter the character's ASCII value or simply type the desired character. Note that this command clears the current character before tracing the pattern of the new one.

The save command (S) saves the current character pattern (the contents of the editing window) to the font file. To add the pattern to the font file, you must save it before moving to another character or exiting from the editor. After you save a pattern, you will automatically be moved to the next character in the set. Note that all the data for the pattern may not be written immediately to disk when you use the S command. The computer collects the data in a buffer and writes to disk only when the buffer is full. For this reason, it's important to always exit the program with the Q (quit) command. If you use Ctrl-Break to break out, the last editing changes you made may not be written to disk.

The initialize command (I) resets the size of the font and allows you to change its call code. If you change the font size, you'll probably need to use the clear all (A) command to erase any previous character definitions. Character patterns designed for one font size will appear garbled when displayed in another size.

The clear all command (A) clears all the characters in the current font. Use this command with care; it's not possible to recover the character patterns once they have

been erased.

The rewrite command (W) erases and redisplays the entire editing screen. The editing window is redrawn with the character pattern from the font file.

The restart command (R) restarts the program. The current chararacter set patterns will be cleared, and you'll be given the opportunity to choose another font file for editing.

Press the Q key to exit the program and return to BASIC. All saved character patterns will be written to disk before the program exits.

The clear all, restart, and quit commands all ask *Are you sure?* before proceeding. Respond with *Y* to execute the command or *N* to cancel the command. You can also cancel the clear, trace, and save commands by pressing the space bar while the command is acting.

The draw-mode command (D) allows you to turn automatic draw mode off and on. When draw mode is on, the cursor automatically turns on every square which it moves over. This is useful for filling large areas of the editing window. When draw mode is off, the cursor moves without disturbing anything in the edit window.

The erase-mode command (E) allows you to turn automatic erase mode off and on. When erase mode is on, the cursor automatically erases every square which it moves over.

Both draw mode and erase mode are canceled when you draw with the space bar.

Creating Headers

A header is simply a picture made of many custom characters. Perhaps the most common use of a header is to create a letterhead which goes at the top of a page of stationery. (See Figure 2.) However, you can use this feature to put graphics anywhere on a page.

The first step in creating a header is to decide on its size. You must subdivide the header into blocks of character size. For example, to create a header that is 240 dots wide by six lines high, you could use character blocks 24 dots wide by three lines high, in which case 20 characters would be required for the header—two rows of 10 characters each. Other character sizes could be used, such as 12 dots wide by two lines high, in which case the header must be subdivided into more blocks. The only restriction is that the complete header pattern can use no more that 94 blocks (only characters 33–126 can be defined). If you use the largest character size (36 dots wide by three lines high) and divide the header into 26 characters across by 3 characters high, you can create a header line that is three times as high as the largest font style and the width of 80 normal characters, a space about 8 × ¾ inches in size.

If you create a header of the maximum size, the three lines of character blocks can be divided in many different ways. For instance, you might use the upper two lines to create a custom letterhead for the top of a page, and use the remaining line to create a design for the bottom of the page.

Header characters can be created by drawing as you go on the editing screen. However, you may find it somewhat difficult to visualize the overall design, since only one character is visible in the editing window. Perhaps the simplest method is to tape together several sheets of graph paper and design the header completely before you begin editing with the font editor. Then decide what portion of the design should go in each character block, and begin filling in the characters. You'll have an easier time remembering which character blocks comprise the header pattern if you use a sequential series of characters for the patern. For example, if you subdivide your header design into two rows of ten characters, you might use characters 65–74 (corresponding to *A–J*) for the top row and characters 75–84 (corresponding to *K–T*) for the bottom row. Be sure you remember which characters you used for your design. You'll need this information to print the header later.

Forbidden Character Values

When you save a character, its pattern is converted into a series of numbers in the range 0–255. Two of the 256 possible values create problems when you attempt to write them to disk as part of a file. The tab character (ASCII 9) is written as five space characters rather than as one tab character. Character 26 signifies the end of a file and prevents all subsequent values from becoming part of the file. If you try to save a pattern containing either of these values, the program shows you which dots in the pattern create the problem number (they will be changed to red), and it gives you a chance to correct them. Simply change one of the offending dots in the vertical column containing a problem pattern and try to save the character again. Fortunately, these values appear infrequently.

Using Custom Fonts

Once you have created a custom font or header, you can use the font or header in a word processor document. The first step, of course, is to create the document and decide which fonts you wish to use. You can use almost any word processor, as long as it has an option to store documents as ASCII text files. (Program 2, the printing program, can use only ASCII text files.) You can use any of the fonts created with the font editor as well as the standard character styles available in your word processor or printer. A few special rules must be followed when preparing a document to use the custom fonts and headers.

Using a font that's wider than the usual 12-dot width may require some extra planning. Because the characters take up more space horizontally, you may have problems centering them or determining how many will fit on a line. For instance, if you are using double-width (24-dot) characters, you should reduce the margins on your word processor by half, or insert an extra space between each character in a line so that the word processor will not attempt to store too many characters per line. If a line of text translates into font characters requiring more than the maximum number of dots the printer can place on a line (960 dots for the IBM Graphics Printer), characters to the right of the limit will be lost. Similarly, you may encounter problems when trying to set the page length for fonts

Increase your knowledge about all aspects of compu

An absolutely no-risk guarantee.

values to $126.75

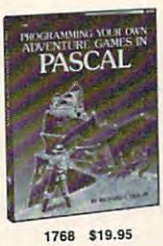
Select 5 Books for only $3.95

More programs, projects, and ways to use your micro.
Keep well-informed about the latest books available—and get the *original publishers' editions* at discounts of up to 50% off the publishers' prices!

2850 $25.95

1884P $18.95

2738 $29.95
Counts as 2



1768 $19.95

1816 $15.95

1876P $17.95

1970 $22.95

1988 $23.95

2732 $18.95

2831 $22.95

1085P $10.25

1807 $17.95

2622 $21.95

2710 $27.95
Counts as 2

1251P $10.25

2623 $21.95

1535 $16.95

1737 $18.95

2761 $19.95

2691 $23.95

1724P $13.50

2756 $18.95

2650 $21.95

2856 $25.95

(Publisher's Prices Shown)

1990 $24.95

2766P $12.95

1275 $15.95

2694 $22.95

Membership Benefits

• **Big Savings.** In addition to this introductory offer, you keep saving substantially with members' prices of up to 50% off the publishers' prices. • **Bonus Books.** Starting immediately, you will be eligible for our Bonus Book Plan, with savings of up to 80% off publishers' prices. • **Club News Bulletins.** 13 times per year you will receive the Book Club News, describing all the current selections—mains, alternates, extras—plus bonus offers and special sales, with hundreds of titles to choose from. • **Automatic Order.** If you want the Main Selection, do nothing and it will be sent to you automatically. If you prefer another selection, or no book at all, simply indicate your choice on the reply form provided. • **Ironclad No-Risk Guarantee.** If not satisfied with your books, return them within 10 days without obligation! • **Exceptional Quality.** All books are quality publishers' editions especially selected by our Editorial Board.

1997 $21.95

2749 $22.95

2730 $27.95
Counts as 2

2771 $25.95

1993 $21.95

2855 $24.95

All books are hardcover editions unless numbers are followed by a P for paperback.

The Computer Book Club®

P.O. Box 80, Blue Ridge Summit, PA 17214

Please accept my membership in The Computer Book Club® and send the 5 volumes circled below, billing me $3.95 plus shipping and handling charges. If not satisfied, I may return the books within ten days without obligation and have my membership canceled. I agree to purchase at least 3 books at regular Club prices (plus shipping/handling) during the next 12 months, and may resign any time thereafter.

1085P 1251P 1275 1535 1724P 1737 1768 1807 1816 1876P 1884P
1970 1988 1990 1993 1997 2622 2623 2650 2691 2694 2710 2730
2732 2738 2749 2756 2761 2766P 2771 2831 2850 2855 2856

Name _____

Address _____

City _____

State/Zip _____ Phone _____

Valid for new members only. Foreign applicants will receive special ordering instructions. Canada *must* remit in U.S. currency. This order subject to acceptance by The Computer Book Club®.

CMPT-587

that are taller than usual—each line printed in the tall font will occupy more than one line printed in standard height. You can compensate by inserting a blank line between each line of double-height text, or two blank lines between each line of triple-height text.

Comma Command

To change the font style within a word processing document, you must include a *comma command* at the point of change. A comma command is simply a comma (,) followed immediately by a the call code of the font which you wish to use. (Remember, use the *call code* for the font, *not* the filename under which the font is stored.) When the document is printed, the comma command tells the printing program (Program 2) which font to use at that point in the printout.

All of the fonts created by the font editor program are *line fonts*, meaning that you must print an entire line of text in the selected font, not just part of the line. The comma to begin the command must be the first nonspace character in the line, except that leading form feed characters, CHR$(12), are allowed. Only one comma command is allowed per line of text.

The comma must be followed immediately (without spaces) by the call code of the desired font, which must be entirely in uppercase characters. A comma command can also take several optional parameters. Here is a list of the comma command options:

S	space following lines horizontally
SS	space following lines horizontally and vertically
D	double strike
G	change printer graphics mode
H	horizontal expansion
V	vertical expansion

You need not include any options in the comma command. For example, if you simply wish to change to the custom font named MYFONT, you would insert this comma command at the beginning of the line where you want the change to take effect:

```
,MYFONT:Your text goes here.
```

Notice that the call code (MYFONT) contains no spaces. The comma command must be separated from the text to be printed by a colon (:). In this case, your text consists of the words *Your text goes here*. If you include options in the comma command, each option must be preceded by a hyphen (-). For instance, this comma command changes the font to MYFONT and causes the printer to double-strike each character one dot below the first character.

```
,MYFONT-D1:Your text goes here.
```

The S and SS options tell the printer program how to handle large fonts. The S option assumes that you have provided extra spaces between each character to be printed; this option is appropriate when you are printing characters that are normal height, but wider than normal. The SS option makes the same assumption about horizontal spacing and further assumes that you have inserted an extra line between each line of text to be printed; this option is appropriate when you are using characters that are both wider and higher than usual.

The D option invokes double-strike mode, in which the printer prints each character, then backs up and prints it again before proceeding to the next character. You may follow the D with a number from 1 to 3 to control how many dots below the first character the second character is printed. Double-strike values of 1 or 2 make characters appear darker than normal. Larger values create a mirrored or doubling effect.

The G option changes the printer's graphics mode. This permits you to squeeze or expand existing fonts even further by invoking built-in printer modes. The G should be followed by the two-digit numeric code of the option you want to invoke. For an IBM Graphics Printer and compatibles, the codes 75, 76, and 90 invoke normal graphics, double-wide graphics, and compressed graphics, respectively. Thus, the comma command **,MYFONT-G90** causes the printer to use the characters in MYFONT, using compressed graphics mode.

The H and V options affect the optional automatic spacing invoked by the S or SS options (see above). The H option is followed by a number in the range 1–9 to indicate how many times the font should be expanded horizontally. The V option is followed by a number from 1 to 4 to indicate how many times to expand the font vertically. For example, the comma command **,MYFONT-H3-V2-SS** tells the printer program to print each subsequent character three times its normal width and two times its usual height.

Expanding characters with the H and V options can be a slow process. To indicate that something is happening, the program flashes an exclamation point (!) on the screen.

Back To Normal

To cancel a custom font and resume printing with the printer's standard character set, insert this comma command:

```
,REGULAR
```

Note that this command cannot use any of the options of the other commands. There are two ways of changing the print style while using the standard character set. The first is to use the usual formatting commands for your word processor. You must use some care, however, when mixing these commands with Font Printer comma commands. To use this method, insert all of the comma commands needed to do what you want, then *print the document to disk* using your word processor's printer option or print program. It is important to include this step so that the output is reformatted according to your embedded formatting commands and so that the final file is in ASCII.

Standard Fonts

The second way to change printer styles is to define special printer font call codes. The definitions must be entered as DATA statements at the end of Program 2. Remember, a standard printer style is one which your printer can print without the aid of Font Printer. Program 2 must know three things for each standard character style: the style's call code, the ASCII code or sequence of codes which invokes the style, and the code or sequence which disables the style.

Definitions for some styles available on the IBM Graphics Printer are already in the DATA

lines at the end of Program 2, but you may want to add more. To avoid confusion, it is best to put each set of standard font information on its own DATA line. Begin by typing a descriptive name for the style. This name is the style's call code; note that the call code must be entirely in uppercase, with no spaces, colons, or hyphens. The call code must be followed by the ASCII value or values which invoke (enable) this font, each number separated by a comma. Next must come the value −1, which marks the end of the invoking sequence. In the same manner, enter the ASCII values which turn off (disable) the font, following that sequence with another −1. Here are two examples for standard compressed and double-width compressed modes:

```
1010 DATA COMPRESSED,15,-1,18,-1
1020 DATA DOUBLECOMPRESSED,
15,14,-1,20,18,-1
```

Notice that two codes are used to enable double-width compressed mode: The first value—equivalent to CHR$(15)—invokes compression and the second (14) invokes double-width printing.

If you add new DATA lines to Program 2, note that the last DATA item must be named ENDD. Do not delete the line containing the REGULAR call code; this item is needed to return to normal print mode after you have invoked a custom font.

Once you have defined the special font codes for the standard styles, you can invoke the styles by including comma commands just like those for the custom fonts. For example, if you have defined a call named COMPRESSED, you can invoke that style with the command **,COMPRESSED:**.

Printing Headers

Headers are printed in much the same manner as any other font created by Font Printer. The comma commands have the same effect for headers as for any font. The only difference is in how the different parts of the picture are written into the document. Remember, a header consists of many different blocks which have been designed to make up one large picture. Thus, your word processing document would contain the constituent characters which, when redefined, make up the picture.

To illustrate, say that you have created a graphic header named MYHEAD using all of the characters from ASCII 33–110; you have used the largest font style as suggested earlier, and the design occupies three lines, 26 characters to each line (characters 33–58 were used for the top line, 59–84 for the middle, and 85–110 for the bottom). In the header file—and in the final printed product—each character's pattern makes up part of the overall design. But here is the way the header would appear in a word processing document before printing to disk:

```
,MYHEAD:!"#$%&'()*+,-./0123456789:
;<=>?@ABCDEFGHIJKLMNOPQRST
UVWXYZ[ \]^_`abcdefghijklmn
```

The comma command **,MYHEAD** tells the printer program to use the font from the file MYHEAD. Like other comma commands, it is separated from subsequent text with a colon. After the comma command comes the series of characters which, when translated by the printer program, creates the graphic design of the header.

Printing A Document

When you run Program 2, it asks you for the name of the file to print. Enter the name of the ASCII text file that contains your document. There are two different ways to print the document. The first is to enter LPT1: (or simply press Enter) when the program asks you for the output file/device. This option causes the document to be printed directly to the printer, a method which works in most cases.

If the first method does not produce the expected results, or if you wish to print more than one copy of the document, use this technique: When prompted for the output file/device, enter a filename. The program creates a disk file containing the data that would otherwise have been sent to the printer. Once saved on disk, the document can be printed in one of two ways. You can use the DOS PRINT command to put the file in the print queue, allowing you to run other programs while the file is printing. You can also use TYPE and redirect the output to the printer instead of the screen. Here are examples of both commands (remember, these are DOS commands which you enter from the DOS prompt):

PRINT *filename*
TYPE *filename* **>LPT1:**

The expression >LPT1: diverts the output from TYPE to the printer. TYPE works about twice as fast as PRINT, but it doesn't allow you to perform other tasks while printing like PRINT does.

After you enter the output file/device, Program 2 asks for the name of the disk drive (be sure to include the colon—A:, B:, and so on) which contains the font files. This allows you to keep your font files on a separate disk. If you have more than one drive, put the document disk (the one containing your text file) in drive A: and the the font disk in drive B:; then specify B: for the drive containing font files. If you have only one disk drive, you can either put the text file on the same disk as the font files, or you can enter B: for the font-file drive. In the latter case, you'll have to repeatedly swap the document and font disks. (When it's time to swap disks, the computer will beep twice. Wait for a message to insert the correct disk.) In any case, the font disk must also contain a font code directory file (FONTCODE.DIR).

If the program can't find a font file called in your document, it indicates the error and gives you the option of inserting a disk containing the specified font file or ignoring the font change.

After you have answered all the necessary questions, Program 2 prints the document to a file or to the printer, according to your choice.

Listing Call Codes

Program 3 helps you keep font files in order. To get a complete list of the font call codes for all of the font files in the current font code directory for a disk, use Program 3. You can direct the listing of font codes and filenames to the screen, printer, or a disk file. Program 3 also has an option to create a file containing a sample of all the fonts in the font code directory. If you choose this option, the program will create a disk file named ALLFONTS, which you can then print with Program 2.

Printing Banners

Program 4 prints banners using custom fonts which you have created with the font editor. After you enter the words to be printed on the banner and the font call code for the font in which the banner is to be printed, the program displays the possible print sizes and asks you how many times to expand the font horizontally and vertically. In most cases, the best results are obtained by selecting a horizontal expansion value that is about half the vertical expansion value. This prints the font with about the same proportions as it would normally have.

After you select the banner size, you are given three different ways to make up each letter. In the first method, the banner letters are made from the words of the message itself. For example, if the message is *Happy Birthday*, each letter is made up of the letters *HappyBirthday*. The second method is to create each character out of normal-sized versions of the character itself. (The large *H* is made of small *H* characters, and so on). The third method lets you choose the character or combination of characters to use for the banner; for instance, to make a happy birthday banner for your friend Bill, you might use *BILL* to make up each character.

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

Program 1: Font Editor

```
FM 10 SCREEN 0,1:WIDTH 80:KEY OF
      F:DIM CHAR(2,37),CODE$(2,5
      0):COL=3
QI 20 CLS:LOCATE 1,14:COLOR 0,5:
      PRINT" Font Editor ":LOCAT
      E 3,1:COLOR 5,0:INPUT"Name
       of font file to edit >",F
      ILE$:B$=FILE$:GOSUB 940:IF
       B$="" THEN BEEP:GOTO 20 E
      LSE FILE$=B$
LJ 30 A$="":ON ERROR GOTO 40:OPE
      N"I",#1,FILE$:CLOSE #1:ON
      ERROR GOTO 0:GOTO 50
HO 40 IF ERR=53 THEN A$="NOT ":R
      ESUME 50 ELSE 980
IL 50 ON ERROR GOTO 60:OPEN"I",#
      3,"FONTCODE.DIR":ON ERROR
      GOTO 0:GOTO 70
BD 60 IF ERR=53 THEN OPEN"O",#3,
      "FONTCODE.DIR":CLOSE #3:RE
      SUME 50 ELSE 980
KE 70 IF NOT(EOF(3)) THEN FOR Z=
      0 TO 2:INPUT #3,CODE$:NEXT
       Z:IF CODE$<>FILE$ THEN 70
       ELSE IF A$="" THEN 90
OA 80 CLOSE #3:GOSUB 570:GOTO 10
      0
BK 90 CLOSE #3:OPEN"R",#1,FILE$,
      4:FIELD #1,2 AS B$,2 AS C$
      :GET #1:X=VAL(B$):Y=VAL(C$
      )*8:CLOSE #1
GL 100 OPEN"R",#1,FILE$,X:FIELD
      #1,X AS D$:CHR=1
PD 110 COLOR 6,0:LOCATE 8,1:PRIN
      T"M = MOVE to new charact
      er":PRINT"C = CLEAR editi
      ng window":PRINT"T = TRAC
      E (copy) image from chara
      cter":PRINT"S = SAVE char
      acter to disk":PRINT"I =
      reINITIALIZE font paramet
      ers":PRINT"A = clear ALL
      characters"
NP 120 PRINT"W = reWRITE screen"
      :PRINT"R = RESTART progra
      m":PRINT"Q = QUIT":PRINT"
      SPACE BAR = plot/erase po
      int"TAB(10)"(or cancel cu
      rrent command)":PRINT"D =
       DRAW is OFF":PRINT"E = E
      RASE is OFF":DRAWS=0:ERAS
      ES=0
IJ 130 GOSUB 920:COLOR 4,0:FOR Z
      =1 TO 25:LOCATE Z,42:PRIN
      T SPACE$(38);:NEXT Z:FOR
      Z=1 TO X:LOCATE Y+1,Z+42:
      PRINT RIGHT$(STR$(Z),1);:
      NEXT Z:FOR Z=0 TO 1:FOR Z
      1=1 TO Y:LOCATE Z1,42+(Z*
      (X+1)):PRINT RIGHT$(STR$(
      Z1),1);:NEXT Z1,Z:XP=1:YP
      =1:CR=CHR
MD 140 GOSUB 890:FOR Z=1 TO Y/8:
      GET #1,((CR-1)*Y/8)+Z+1:F
      OR Z1=1 TO X:A=ASC(MID$(D
      $,Z1,1)):IF A=0 THEN 180
LM 150 IF INKEY$=" " THEN GOSUB
      930:GOTO 190
GH 160 FOR Z2=7 TO 0 STEP -1:IF
      A=>2^Z2 THEN A=A-(2^Z2):L
      OCATE (Z-1)*8+(8-Z2),Z1+4
      2:COLOR 5,5:PRINT" ";
KF 170 NEXT Z2
LM 180 NEXT Z1,Z:COL=(SCREEN(YP,
      42+XP,1)AND 15)
HB 190 COLOR ,0:LOCATE 23,1:PRIN
      T SPACE$(40);
PE 200 LOCATE YP,XP+42:COLOR 1,C
      OL:PRINT CHR$(1);
MO 210 A$=INKEY$:IF A$="" THEN 2
      10
NE 220 IF CANCEL THEN CANCEL=0:C
      OLOR ,0:LOCATE 21,1:PRINT
       SPACE$(40);
JJ 230 IF LEN(A$)=2 THEN 390 ELS
      E IF A$=" " THEN IF DRAWS
       THEN A$="D" ELSE IF ERAS
      ES THEN A$="E" ELSE COLOR
       5,5:LOCATE YP,XP+42:PRIN
      T" ";:IF COL=5 OR COL=4 T
      HEN COL=3:GOTO 200 ELSE C
      OL=5:GOTO 200
PA 240 LOCATE YP,XP+42:COLOR COL
      ,COL:PRINT" ";:A$=CHR$(AS
      C(A$)+32*(A$>="a" AND A$<
      ="z"))
MF 250 IF A$="C" THEN COL=3:GOSU
      B 890:GOTO 200
EO 260 IF A$="T" THEN 500
FK 270 IF A$="S" THEN 420
JN 280 IF A$="W" THEN GOSUB 560:
      GOTO 110
DP 290 IF A$="M" THEN 530
EI 300 IF A$="I" THEN GOSUB 960:
      CLOSE #1:GOSUB 720:GOTO 1
      30
JH 310 IF A$="A" THEN GOSUB 960:
      GOSUB 860:GOSUB 890:COL=3
      :CHR=1:GOSUB 920:GOTO 200
GB 320 IF A$="R" THEN GOSUB 960:
      CLOSE #1:RUN
PL 330 IF A$="Q" THEN GOSUB 960:
      CLOSE #1:SCREEN 0:CLS:END
FI 340 IF A$="D" THEN DRAWS=1-DR
      AWS:COL=5:IF ERASES THEN
      ERASES=0
GI 350 IF A$="E" THEN ERASES=1-E
      RASES:COL=3:IF DRAWS THEN
       DRAWS=0
HD 360 LOCATE 19,13:IF DRAWS THE
      N COLOR 14,0:PRINT"ON " E
      LSE COLOR 6,0:PRINT"OFF"
QI 370 LOCATE 20,14:IF ERASES TH
      EN COLOR 14,0:PRINT"ON "
      ELSE COLOR 6,0:PRINT"OFF"
CI 380 GOTO 200
LK 390 A$=RIGHT$(A$,1):XO=XP:YO=
      YP:IF A$="H" AND YP>1 THE
      N YP=YP-1 ELSE IF A$="M"
      AND XP<X THEN XP=XP+1 ELS
      E IF A$="P" AND YP<Y THEN
       YP=YP+1 ELSE IF A$="K" A
      ND XP>1 THEN XP=XP-1 ELSE
       200
DM 400 LOCATE YO,XO+42:COLOR COL
      ,COL:PRINT" ";:COL=(SCREE
      N(YP,XP+42,1)AND 15):IF D
      RAWS THEN COL=5 ELSE IF E
      RASES THEN COL=3
IM 410 LOCATE YP,XP+42:COLOR 1,C
      OL:PRINT CHR$(1);:GOTO 20
      0
PO 420 SAVED=0:COLOR 2,0:LOCATE
      21,1:PRINT"SAVING"SPACE$(
      18);
PD 430 FOR X1=43 TO X+42:FOR NUM
      =1 TO Y/8:BYTE=0:FOR Y1=8
       TO 1 STEP -1:BYTE=BYTE-2
      ^(8-Y1)*((SCREEN(Y1+(NUM-
      1)*8,X1,1)AND 15)<>3):NEX
      T Y1
HM 440 IF INKEY$=" " THEN GOSUB
      930:GOTO 200
KO 450 CHAR(NUM-1,X1-42)=BYTE:IF
       BYTE=9 THEN COLOR 4,4:LO
      CATE (NUM-1)*8+5,X1:PRINT
      " ";:LOCATE (NUM-1)*8+8,X
      1:PRINT" ";:SAVED=1
GL 460 IF BYTE=26 THEN COLOR 4,4
      :LOCATE (NUM-1)*8+4,X1:PR
      INT" ";:LOCATE (NUM-1)*8+
      5,X1:PRINT" ";:LOCATE (NU
      M-1)*8+7,X1:PRINT" ";:SAV
      ED=1
CC 470 NEXT NUM,X1:IF SAVED=1 TH
      EN COLOR 4,0:LOCATE 21,1:
      PRINT"PATTERN CANNOT BE S
      AVED":CANCEL=1:GOTO 200
MK 480 FOR Z=1 TO Y/8:B$="":FOR
      Z1=1 TO X:B$=B$+CHR$(CHAR
      (Z-1,Z1)):NEXT Z1:LSET D$
      =B$:PUT #1,((CHR-1)*Y/8)+
      Z+1:NEXT Z
FC 490 CHR=CHR-1*(CHR<94):GOSUB
      920:COLOR ,0:LOCATE 21,1:
      PRINT SPACE$(6):CR=CHR:GO
      TO 140
FE 500 COLOR 3,0:LOCATE 23,1:LIN
      E INPUT"Character to trac
      e (!-~ or 33-126) >",B$:I
      F B$="" THEN 190
MP 510 CR=VAL(B$)-32:IF CR<=-23
      THEN IF B$>"~" OR B$<"!"
      THEN BEEP:GOTO 500 ELSE C
      HR=ASC(B$)-32:GOSUB 920:C
      R=CHR:GOTO 140 ELSE IF CR
      <1 OR CR>94 THEN BEEP:GOT
      O 500
DA 520 GOTO 140
KL 530 COLOR 3,0:LOCATE 23,1:LIN
      E INPUT"Character to edit
       (!-~ or 33-126) >",B$:IF
       B$="" THEN 190
```

```
HE 540 CR=VAL(B$)-32:IF CR<=-23
       THEN IF B$>"~" OR B$<"!"
       THEN BEEP:GOTO 530 ELSE C
       HR=ASC(B$)-32:GOSUB 920:C
       R=CHR:GOTO 140 ELSE IF CR
       <1 OR CR>94 THEN BEEP:GOT
       O 530
JM 550 CHR=CR:GOSUB 920:GOTO 140
IF 560 CLS:LOCATE 1,14:COLOR 0,5
       :PRINT" Font Editor ":LOC
       ATE 3,1:COLOR 5,0:PRINT"C
       urrent font filename >"FI
       LE$:RETURN
CK 570 COLOR 12:LOCATE 21,1:IF C
       ODE$<>FILE$ THEN 590
DN 580 PRINT FILE$;" appears in
       the font code directory,
       but isn't on disk.":PRINT
       "Do you wish to create a
       new version of ";FILE$;"
       (Y/N)? ";:GOTO 600
FB 590 PRINT"There is no entry f
       or ";FILE$;" in the font
       code directory.":PRINT"Th
       ere is ";A$;"a file named
        ";FILE$;" on this disk."
       :PRINT"Do you wish to cre
       ate an entry for ";FILE$;
       " (Y/N)? ";
LD 600 B$=INPUT$(1):IF B$="N" OR
        B$="n" THEN RETURN 20
NC 610 IF B$<>"Y" AND B$<>"y" TH
       EN BEEP:GOTO 600
KC 620 LOCATE 21,1:FOR Z=1 TO 3:
       PRINT SPACE$(70):NEXT Z:L
       OCATE 21,1:PRINT"(E) = st
       art with an EMPTY font":P
       RINT"(C) = start with a C
       OPY of an existing font":
       IF A$="" THEN PRINT"(A) =
        ADD an existing font fil
       e to the font code direct
       ory"
FC 630 B$=INPUT$(1):IF B$="E" OR
        B$="e" THEN GOSUB 560:A$
       ="NEW":GOTO 720
NH 640 IF B$="A" OR B$="a" THEN
       IF A$="" THEN SOURCE$=FIL
       E$:GOTO 670
CK 650 IF B$<>"C" AND B$<>"c" TH
       EN BEEP:GOTO 630
OA 660 CLS:INPUT"Filename of fon
       t to copy >",SOURCE$:B$=S
       OURCE$:GOSUB 940:IF B$=""
        THEN BEEP:GOTO 660 ELSE
       SOURCE$=B$
JB 670 ON ERROR GOTO 710:OPEN"I"
       ,#1,SOURCE$:CLOSE #1:OPEN
       "R",#1,SOURCE$,4:FIELD #1
       ,2 AS B$,2 AS C$:GET #1:X
       =VAL(B$):Y=VAL(C$)*8:CLOS
       E #1:ON ERROR GOTO 0
GI 680 IF SOURCE$=FILE$ THEN GOS
       UB 560:GOTO 780
NI 690 OPEN"R",#1,SOURCE$,X:FIEL
       D #1,X AS B$:OPEN"R",#2,F
       ILE$,X:FIELD #2, X AS C$
IE 700 FOR Z=1 TO 94:GET #1,Z+1:
       LSET C$=B$:PUT #2,Z+1:NEX
       T Z:CLOSE:GOSUB 560:GOTO
       780
IL 710 GOSUB 560:LOCATE 5,1:PRIN
       T"ERROR: ";SOURCE$;" not
       found or couldn't be read
       .":RESUME 620
EM 720 COLOR 5,0:LOCATE 4,1:INPU
       T"Character width in dots
        (4-36) > ",X:IF X<4 OR X
       >36 THEN BEEP:GOTO 720
BJ 730 LOCATE 5,1:INPUT"Characte
       r height in lines (1-3) >
        ",Y:Y=Y*8:IF Y<1 OR Y/8>
       3 THEN BEEP:GOTO 730
EM 740 OPEN"R",#1,FILE$,4:FIELD
       #1,2 AS B$,2 AS C$
PL 750 LSET B$=RIGHT$(STR$(X),2)
       :LSET C$=RIGHT$(STR$(Y/8)
       ,2):PUT #1,1:CLOSE #1
EH 760 IF A$="I" OR A$="NEW" THE
       N OPEN"R",#1,FILE$,X:FIEL
       D #1,X AS D$
HO 770 IF A$="NEW" THEN GOSUB 86
       0:CLOSE #1
JJ 780 COLOR 3,0:LOCATE 23,1:INP
       UT"Enter code name for th
       is font >",CODE$:B$=CODE$
       :GOSUB 940:IF B$="" THEN
       BEEP:GOTO 780 ELSE CODE$=
       B$
AC 790 LOCATE 23,1:INPUT"Font ty
       pe: (C) = Character or (H
       ) = Header  >",TYPE$:IF T
       YPE$="C" OR TYPE$="c" THE
       N TYPE=0 ELSE IF TYPE$="H
       " OR TYPE$="h" THEN TYPE=
       2 ELSEBEEP:GOTO 790
BF 800 OPEN"I",#3,"FONTCODE.DIR"
       :Z=0
FK 810 WHILE NOT(EOF(3)):FOR Z1=
       0 TO 2:INPUT #3,CODE$(Z1,
       Z):NEXT Z1:Z=Z+1:WEND:CLO
       SE #3:Z1=0
KB 820 IF Z1=Z THEN Z=Z+1 ELSE I
       F CODE$(2,Z1)<>FILE$ THEN
        Z1=Z1+1:GOTO 820
EN 830 CODE$(0,Z1)=STR$(TYPE):CO
       DE$(1,Z1)=CODE$:CODE$(2,Z
       1)=FILE$
BL 840 OPEN"O",#3,"FONTCODE.DIR"
       :FOR Z1=0 TO Z-1:FOR Z2=0
        TO 2:PRINT #3,CODE$(Z2,Z
       1):NEXT Z2,Z1:CLOSE #3
JP 850 COLOR ,0:LOCATE 23,1:PRIN
       T SPACE$(50);:RETURN
HG 860 FOR Z=1 TO 94:FOR Z1=1 TO
        Y/8
OP 870 IF INKEY$=" " AND A$="A"
       THEN 930
KF 880 LSET D$=STRING$(X,CHR$(0)
       ):PUT #1,(Z-1)*Y/8+Z1+1:N
       EXT Z1,Z:RETURN
KM 890 FOR Z1=1 TO Y:COLOR 3,3:L
       OCATE Z1,43:PRINT SPACE$(
       X);
BI 900 IF INKEY$=" " AND A$="C"
       THEN 930
DB 910 NEXT Z1:Z1=FRE("c"):RETUR
       N
LF 920 COLOR 7,0:LOCATE 6,1:PRIN
       T"Current character = ";:
       COLOR 15:PRINT CHR$(CHR+3
       2);:COLOR 7:PRINT SPC(5);
       "ASCII =";:COLOR 15:PRINT
        CHR+32;:RETURN
OM 930 COL=(SCREEN(YP,42+XP,1)AN
       D 15):CANCEL=1:COLOR 4,0:
        LOCATE 21,1:PRINT"COMMAND
        CANCELLED":WHILE INKEY$<
       >"":WEND:RETURN
JI 940 C$="":FOR Z2=1 TO LEN(B$)
       :ASCII=ASC(MID$(B$,Z2,1))
       :IF ASCII<>32 THEN C$=C$+
       CHR$(ASCII+32*((ASCII>96)
       AND(ASCII<123)))
FI 950 NEXT Z2:B$=C$:RETURN
IM 960 COLOR 12,0:LOCATE 23,1:PR
       INT"Are you sure you wish
        to "A$" (Y/N)? ";:B$=INP
       UT$(1):IF B$="Y" OR B$="y
       " THEN LOCATE 23,1:PRINT
       SPACE$(41);:RETURN
KJ 970 IF B$="N" OR B$="n" THEN
       RETURN 190 ELSE 960
IC 980 CLOSE:PRINT:PRINT "Error"
       ;ERR;"in line";ERL:RESUME
        990
ME 990 END
```

Program 2: Printing Program

```
IF 10 SCREEN 0:WIDTH 80:COLOR 2,
      0,0:KEY OFF:DIM TEXT$(94,2
      ),CODE$(3,100),B#(960)
QO 20 READ A$:IF A$<>"ENDD" THEN
       CODE$(0,Z)="1":CODE$(1,Z)
      =A$:FOR Z1=2 TO 3:READ A:W
      HILE A<>-1:CODE$(Z1,Z)=COD
      E$(Z1,Z)+CHR$(A):READ A:WE
      ND:NEXT Z1:Z=Z+1:GOTO 20
NH 30 STANDARD=Z-1
KB 40 CLS:COLOR 0,2:LOCATE 1,33:
      PRINT" Font Printer ":COLO
      R 2,0:LOCATE 3,1:INPUT"Dri
      ve with disk containing do
      cument file (default = A:)
       > ",DISK1$:IF DISK1$="" T
      HEN DISK1$="A:" ELSE IF RI
      GHT$(DISK1$,1)<>":" THEN B
      EEP:GOTO 40
PP 50 LOCATE 5,1:PRINT"Insert di
      sk containing document fil
      e into drive ";DISK1$
GF 60 LOCATE 7,1:PRINT SPACE$(78
      ):LOCATE 7,1:INPUT"Name of
       ASCII document file to pr
      int > ",IN$:IF IN$="" THEN
       BEEP:GOTO 60
DL 70 ON ERROR GOTO 820:OPEN"I",
      #2,DISK1$+IN$:ON ERROR GOT
      O 0
MP 80 FEND=0:WHILE NOT(EOF(2)):L
      INE INPUT #2,A$:FEND=FEND+
      1:WEND:CLOSE #2
II 90 IF FEND=0 THEN BEEP:LOCATE
       9,1:PRINT"ERROR: Input fi
      le ";IN$;" is empty.":GOSU
      B 760:GOTO 40
CO 100 LOCATE 9,1:PRINT SPACE$(7
       8):LOCATE 9,1:INPUT"Name
       of output file or device
       (default = LPT1:) > ",OUT
       T$:IF OUTT$="" THEN OUTT$
       ="LPT1:"
BB 110 ON ERROR GOTO 830:OPEN"O"
       ,#1,OUTT$:ON ERROR GOTO 0
ED 120 LOCATE 11,1:PRINT SPACE$(
       78):LOCATE 11,1:INPUT"Dri
       ve with disk containing f
       ont files (default = B:)
       > ",DISK2$:IF DISK2$="" T
       HEN DISK2$="B:" ELSE IF R
       IGHT$(DISK2$,1)<>":" THEN
        BEEP:GOTO 120
ML 130 IF DISK2$<>DISK1$ THEN LO
       CATE 13,1:PRINT"Insert di
       sk containing font files
       into drive ";DISK2$
CB 140 PRINT:PRINT STRING$(78,"-
       ")
DN 150 OPEN"R",#2,DISK1$+IN$,1:F
       IELD #2,1 AS I$:FLIN=0:CH
       AR=0:NUM=0
KM 160 FLIN=FLIN+1:IF FLIN<=FEND
        THEN 200
DE 170 CLOSE:PRINT STRING$(78,"-
       "):PRINT"Finished printin
       g ";IN$:PRINT
BN 180 PRINT"Print another docum
       ent (Y/N) > ";:A$=INPUT$(
       1):IF A$="y" OR A$="Y" TH
       EN 40
MO 190 IF A$<>"n" AND A$<>"N" TH
       EN BEEP:GOTO 180 ELSE CLS
       :END
EE 200 A$="":ON ERROR GOTO 880
IO 210 IF EOF(2) THEN 170 ELSE C
       HAR=CHAR+1:GET #2,CHAR:IF
```

```
      ASC(I$)<>13 THEN A$=A$+I
      $:LS=(I$<>" "):GOTO 210
IJ 220 GET #2:IF ASC(I$)<>10 THE
      N LF=0 ELSE CHAR=CHAR+1:L
      F=-1
BO 230 IF INSTR(A$,CHR$(12))=1 T
      HEN A$=RIGHT$(A$,LEN(A$)-
      1):PRINT #1,CHR$(12);
MD 240 Z=INSTR(A$,","):IF Z=0 TH
      EN 540 ELSE IF Z=1 THEN 2
      60
EA 250 FOR Z1=1 TO Z-1:IF MID$(A
      $,Z1,1)<>" " THEN Z=0:NEX
      T Z1:IF Z=0 THEN 540
GE 260 Z1=INSTR(Z,A$,":"):IF Z1=
      0 THEN 540 ELSE FONT$=MID
      $(A$,Z+1,Z1-Z-1)
JL 270 Z2=INSTR(FONT$,"-"):IF Z2
       THEN OPT$=MID$(FONT$,Z2)
      :FONT$=LEFT$(FONT$,Z2-1)
JG 280 COLOR 6:PRINT"font=";FONT
      $;:IF Z2 THEN PRINT" & ";
      OPT$;
BJ 290 IF FONT$=OLD$ THEN 390
MC 300 Z3=0
EM 310 IF Z3<=STANDARD THEN IF C
      ODE$(1,Z3)=FONT$ THEN 380
       ELSE Z3=Z3+1:GOTO 310
GL 320 FLAG=1:GOSUB 780:ON ERROR
       GOTO 860:OPEN"I",#3,DISK
      2$+"FONTCODE.DIR":ON ERRO
      R GOTO 0
OD 330 Z=STANDARD+1:WHILE NOT(EO
      F(3)):FOR Z3=0 TO 2:INPUT
       #3,CODE$(Z3,Z):NEXT Z3:Z
      =Z+1:WEND:CLOSE #3:Z3=STA
      NDARD+1
MJ 340 IF Z3<Z THEN IF CODE$(1,Z
      3)=FONT$ THEN 380 ELSE Z3
      =Z3+1:GOTO 340
DM 350 BEEP:PRINT:PRINT:PRINT"No
       entry for ";FONT$;" in f
      ont code directory."
FE 360 PRINT"(I) = Ignore font c
      hange":PRINT"(R) = Retry
       on another font file disk
      " :B$=INPUT$(1):IF B$="i"
       OR B$="I" THEN FLAG=2:GO
      SUB 780:GOTO 530
NH 370 IF B$="r" OR B$="R" THEN
       320 ELSE BEEP:GOTO 360
DO 380 NUM=Z3:IF VAL(CODE$(0,NUM
      ))=1 THEN PRINT:GOTO 530
KE 390 DOUBLE=0:DNUM=0:GR=76:SPA
      CED=0:VERT=1:HOR=1:SP=0:P
      R=0:IF Z2=0 THEN 470
CJ 400 FOR Z3=1 TO LEN(OPT$):B$=
      MID$(OPT$,Z3,1)
JG 410 IF B$="S" THEN SPACED=1:I
      F MID$(OPT$,Z3+1,1)="S" T
      HEN SPACED=2:Z3=Z3+1
IH 420 IF B$="D" THEN DOUBLE=1:D
      NUM=VAL(MID$(OPT$,Z3+1,1)
      ):Z3=Z3+1
KD 430 IF B$="G" THEN GR=VAL(MID
      $(OPT$,Z3+1,2)):Z3=Z3+2
FA 440 IF B$="V" THEN VERT=VAL(M
      ID$(OPT$,Z3+1,1)):Z3=Z3+1
      :IF VERT>4 THEN VERT=4 EL
      SE IF VERT<1 THEN VERT=1
HL 450 IF B$="H" THEN HOR=VAL(MI
      D$(OPT$,Z3+1,1)):Z3=Z3+1:
      IF HOR<1 THEN HOR=1
LB 460 NEXT Z3
HE 470 PRINT #1,CHR$(27)+CHR$(50
      );:IF FONT$=OLD$ THEN NUM
      =ONUM:PRINT " - Font alre
      ady in memory.":GOTO 530
ED 480 ON ERROR GOTO 870:OPEN"I"
      ,#3,DISK2$+CODE$(2,NUM):C
      LOSE #3:ON ERROR GOTO 0
IF 490 OPEN"R",#3,DISK2$+CODE$(2
      ,NUM),4:FIELD #3,2 AS B$,
      2 AS C$:GET #3,1:WIDE=VAL
      (B$):HIGH=VAL(C$):CLOSE #
      3:SP$=STRING$(WIDE,0)
FB 500 OPEN"R",#3,DISK2$+CODE$(2
      ,NUM),WIDE:FIELD #3,WIDE
      AS B$
ND 510 FOR Z=1 TO 94:FOR Z3=0 TO
       HIGH-1:GET #3,(Z-1)*HIGH
      +Z3+2:TEXT$(Z,Z3)=B$:NEXT
       Z3,Z:CLOSE #3:FOR Z=0 TO
       HIGH-1:TEXT$(0,Z)=SP$:NE
      XT Z
BI 520 OLD$=FONT$:ONUM=NUM:PRINT
      "  high=";HIGH;"  wide=";
      WIDE:FLAG=2:GOSUB 780
BH 530 A$=MID$(A$,Z1+1)
PO 540 COLOR 7:IF VAL(CODE$(0,NU
      M))=1 THEN PRINT A$:PRINT
       #1,CODE$(2,NUM);A$;CODE$
      (3,NUM);:IF NOT(LF) THEN
      PRINT #1,CHR$(27);CHR$(74
      );CHR$(1);:GOTO 160 ELSE
      PRINT #1,"":GOTO 160
EI 550 IF SPACED=0 THEN 600
GC 560 LE=LEN(A$):B$=A$:A$="":FO
      R Z=1 TO LE STEP WIDE*HOR
      /12:FOR Z1=1 TO WIDE*HOR/
      12:IF MID$(B$,Z+Z1-1,1)="
       " THEN NEXT Z1:A$=A$+" "
       ELSE A$=A$+MID$(B$,Z+Z1-
      1,1)
FH 570 NEXT Z:IF SPACED=1 THEN 6
      00
HB 580 SP=SP+1:IF SP=HIGH*VERT T
      HEN SP=0:IF NOT(PR) THEN
      600 ELSE PR=0:GOTO 160
HK 590 IF NOT(LS) OR PR THEN 160
       ELSE PR=-1
DD 600 A$=LEFT$(A$,INT(960/(WIDE
      *HOR))):LE=LEN(A$):PRINT
      A$
HF 610 FOR Z1=0 TO HIGH-1:FOR Z2
      =0 TO DOUBLE:IF Z2=1 THEN
       PRINT #1,CHR$(27)+CHR$(5
      1)+CHR$(DNUM+1) ELSE IF Z
      1>0 AND VERT=1 THEN PRINT
       #1,CHR$(27)+CHR$(49)
DB 620 C$=CHR$(27)+CHR$(GR)+CHR$
      ((LE*WIDE*HOR)MOD 256)+CH
      R$(FIX(LE*WIDE*HOR/256)):
      IF VERT=1 THEN PRINT #1,C
      $;
GK 630 FOR Z3=1 TO LE:ASCII=ASC(
      MID$(A$,Z3,1))-32:IF ASCI
      I<0 OR ASCII>94 THEN ASCI
      I=0
IO 640 IF VERT>1 THEN 680
HF 650 IF ASCII=0 THEN FOR Z4=1
       TO HOR:PRINT #1,SP$;:NEXT
       Z4:GOTO 710
AG 660 IF HOR=1 THEN PRINT #1,TE
      XT$(ASCII,Z1);:GOTO 710
OE 670 FOR Z4=1 TO WIDE:A$=MID$(
      TEXT$(ASCII,Z1),Z4,1):FOR
       Z5=1 TO HOR:PRINT #1,A$;
      :NEXT Z5,Z4:GOTO 710
CE 680 FOR Z4=1 TO WIDE:A=ASC(MI
      D$(TEXT$(ASCII,Z1),Z4,1))
      :B#=0
JF 690 FOR Z5=7 TO 0 STEP-1:IF A
      >=2^Z5 THEN A=A-(2^Z5):FO
      R Z6=0 TO VERT-1:B#=B#+(2
      ^Z5^VERT)*(2^Z6):NEXT Z6
NM 700 NEXT Z5:FOR Z5=1 TO HOR:B
      #((Z3-1)*WIDE*HOR+(Z4-1)*
      HOR+Z5)=B#:GOSUB 750:NEXT
       Z5,Z4
FK 710 NEXT Z3,Z2:IF VERT=1 THEN
       730
CG 720 FOR Z2=VERT TO 1 STEP-1:P
      RINT #1,C$;:FOR Z3=1 TO W
      IDE*HOR*LE:PART=INT(((B#(
      Z3)/256^Z2-INT(B#(Z3)/256
      ^Z2))*256^Z2)/256^(Z2-1))
      :PRINT #1,CHR$(PART);:GOS
      UB 750:NEXT Z3:PRINT #1,C
      HR$(27)+CHR$(49):NEXT Z2
BC 730 NEXT Z1:IF VAL(CODE$(0,NU
      M))=2 THEN PRINT #1,"":GO
      TO 160
MC 740 PRINT #1,CHR$(27)+CHR$(50
      ):GOTO 160
PC 750 PRINT"!"CHR$(29)" "CHR$(2
      9);:RETURN
GC 760 PRINT TAB(8)"Press any ke
      y to continue...":WHILE I
      NKEY$="":WEND:RETURN
HD 770 LOCATE 11,1:PRINT SPACE$(
      78):PRINT SPACE$(78):RETU
      RN
HI 780 IF DISK1$<>DISK2$ THEN RE
      TURN ELSE BEEP:ON FLAG GO
      TO 790,800
NA 790 PRINT:PRINT"Remove docume
      nt disk from drive ";DISK
      1$;". Insert font disk.":
      GOTO 810
FD 800 PRINT:PRINT"Remove font d
      isk from drive ";DISK1$;"
      . Insert document disk."
DP 810 BEEP:PRINT"Press any key
      when ready.":WHILE INKEY$
      ="":WEND:RETURN
CB 820 IF ERR<>53 THEN 890 ELSE
      BEEP:LOCATE 9,1:PRINT"ERR
      OR: Input file ";IN$;" no
      t found on the disk in dr
      ive ";DISK1$:GOSUB 760:RE
      SUME 40
MA 830 IF ERR=24 OR ERR=25 OR ER
      R=27 OR ERR=64 OR ERR=68
      THEN BEEP:LOCATE 11,1:PRI
      NT"ERROR #";ERR;"- Check
      device ";OUTT$:GOSUB 760:
      GOSUB 770:RESUME 100
OC 840 IF ERR<>58 THEN 890 ELSE
      BEEP:LOCATE 11,1:PRINT"ER
      ROR: The disk already con
      tains a file named ";OUTT
      $:PRINT"Do you want to re
      place the existing file (
      Y/N) > ";:A$=INPUT$(1):IF
       A$="y" OR A$="Y" THEN KI
      LL OUTT$:GOSUB 770:RESUME
       110
JH 850 IF A$<>"n" AND A$<>"N" TH
      EN BEEP:GOTO 840 ELSE GOS
      UB 770:RESUME 100
QJ 860 IF ERR<>53 THEN 890 ELSE
      BEEP:PRINT:PRINT:PRINT"ER
      ROR: The disk in drive ";
      DISK2$;" has no font code
       directory file.":PRINT T
      AB(8)"Insert another disk
      .":GOSUB 760:RESUME 320
BH 870 IF ERR<>53 THEN 890 ELSE
      BEEP:PRINT:PRINT:PRINT"ER
      ROR: Font file ";CODE$(2,
      NUM);" not found on the d
      isk in drive ";DISK2$:RES
      UME 360
CB 880 IF ERR=15 THEN RESUME 200
CD 890 PRINT:PRINT"Error";ERR;"i
      n line";ERL:RESUME 900
LC 900 END
CG 1000 DATA REGULAR,-1,-1
JP 1010 DATA CONDENSED,15,-1,18,
       -1
EE 1020 DATA DOUBLEWIDTH,14,-1,2
       0,-1
ON 1030 DATA ENDD
```

Program 3: Call Code Lister

```
AG 10 SCREEN 0:COLOR 12,0,0:WIDT
      H 80:CLS
CP 20 PRINT:INPUT"Drive containi
      ng font files (default = A
      :) > ",DISK$:IF DISK$="" T
      HEN DISK$="A:"
PB 30 ON ERROR GOTO 100:OPEN"I",
      #1,DISK$+"FONTCODE.DIR":ON
       ERROR GOTO 0
GP 40 PRINT:PRINT"Select output
      device (P)rinter (S)creen
      or (D)isk" :PRINT
EN 50 A$=INPUT$(1):IF A$="s" OR
      A$="S" THEN OUTT$="SCRN:"
      ELSE IF A$="p" OR A$="P" T
      HEN OUTT$="LPT1:" ELSE IF
      A$="d" OR A$="D" THEN 70 E
      LSE BEEP:GOTO 50
GJ 60 TB=0:OPEN"O",#2,OUTT$:WHIL
      E NOT(EOF(1)):INPUT #1,A$:
      INPUT #1,A$:INPUT #1,B$:PR
      INT #2,TAB(40*TB)B$;" = ";
      A$;:TB=1-TB:WEND:PRINT #2,
      "":CLOSE:END
GD 70 PRINT:PRINT"Do you want th
      e alphabet for each font p
      rinted also (Y/N) > ":A$=I
      NPUT$(1):IF A$="n" OR A$="
      N" THEN 90 ELSE IF A$<>"y"
       AND A$<>"Y" THEN 70
GP 80 C$=CHR$(13)+CHR$(10):D$=C$
      :FOR Z=65 TO 90:C$=C$+CHR$
      (Z):D$=D$+CHR$(Z+32):NEXT
      Z
NN 90 OPEN"O",#2,"ALLFONTS":WHIL
      E NOT(EOF(1)):INPUT #1,A$:
      INPUT #1,A$:PRINT #2,",";A
      $;":";A$;C$;D$:INPUT #1,A$
      :WEND:CLOSE:END
DL 100 BEEP:PRINT:PRINT"The disk
       in drive ";DISK$;" has n
      o font code directory fil
      e.":RESUME 20
```

Program 4: Banner Printer

```
CM 10 SCREEN 0:WIDTH 80:COLOR 1,
      0,0:CLS:DIM TEXT$(94,2)
FC 20 LINE INPUT"Enter Banner wo
      rds > ",A$
JG 30 INPUT"Enter Font Call Code
       > ",CODE$:FOR Z=1 TO LEN(
      CODE$):B$=MID$(CODE$,Z,1):
      IF B$>"`" AND B$<"{" THEN
      C$=C$+CHR$(ASC(B$)-32) ELS
      E IF B$<>" " THEN C$=C$+B$
BL 40 NEXT Z:COLOR 6:PRINT "Inse
      rt disk with font files in
      to disk A: and press any k
      ey when ready":AN$=INPUT$(
      1)
ON 50 CODE$=C$:ON ERROR GOTO 240
      :OPEN"I",#1,"FONTCODE.DIR"
      :WHILE NOT(EOF(1)):INPUT #
      1,C$:INPUT #1,C$:IF CODE$<
      >C$ THEN INPUT #1,C$:WEND:
      CLOSE:BEEP:COLOR 7:LOCATE
      3,1:PRINT "FONT NOT FOUND"
      SPACE$(65):C$="":COLOR 1:L
      OCATE 2,1:GOTO 30
FD 60 INPUT #1,FILE$:CLOSE:LOCAT
      E 3,1:PRINT SPACE$(79):ON
      ERROR GOTO 0:OPEN"R",#1,FI
      LE$,4:FIELD #1,2 AS B$,2 A
      S C$:GET #1,1:WIDE=VAL(B$)
      :HIGH=VAL(C$):CLOSE:OPEN"R
      ",#1,FILE$,WIDE:FIELD #1,W
      IDE AS B$
PH 70 FOR Z=1 TO 94:FOR Z1=0 TO
      HIGH-1:GET #1,(Z-1)*HIGH+Z
      1+2:TEXT$(Z,Z1)=B$:NEXT Z1
      ,Z:CLOSE:SP$=STRING$(WIDE,
      0):FOR Z=0 TO HIGH-1:TEXT$
      (0,Z)=SP$:NEXT Z
FM 80 COLOR 2:LOCATE 4,1:PRINT:F
      OR Z=1 TO 80/(HIGH*8):PRIN
      T USING"&###\\#.##&";SPACE$(
      10);Z;")";Z*8*HIGH/10;" in
      ches tall":NEXT Z:PRINT
BH 90 INPUT"Enter VERTICAL expan
      sion multiple > ",VERT:IF
      VERT=>Z THEN 90
JF 100 COLOR 3:FOR Z1=1 TO Z-1:L
      OCATE Z1+4,50:PRINT USING
       "##\\###.##&";Z1;")";LEN(
      A$)*Z1*WIDE/10.2857;" inc
      hes long":NEXT Z1
GJ 110 LOCATE 5+Z,40:INPUT"Enter
       HORIZONTAL expansion mul
      tiple > ",HOR
PB 120 B$="":FOR Z=1 TO LEN(A$):
      IF MID$(A$,Z,1)<>" " THEN
       B$=B$+MID$(A$,Z,1)
JD 130 NEXT Z:COLOR 6:LOCATE 18,
      25:PRINT"Create the lette
      rs of the banner with:"TA
      B(25)"1) The original str
      ing "TAB(25)"2) Each lett
      er creating itselt"TAB(25
      )"3) You enter the string
       used":PRINT
PL 140 LOCATE 22,30:INPUT"Enter
      selection > ",AN:IF AN>3
      OR AN<1 THEN 140 ELSE IF
      AN=3 THEN LOCATE 23,30:IN
      PUT"Enter String > ",B$
FE 150 COLOR 20:LOCATE 3,18:PRIN
      T"SET UP PRINTER AND PRES
      S ANY KEY WHEN READY":AN$
      =INPUT$(1):LOCATE 3,18:PR
      INT SPACE$(18)"PRINTING"S
      PACE$(18)
PO 160 TABB$=SPACE$(INT((80-VERT
      *8*HIGH)/2)):LPRINT CHR$(
      27);CHR$(49)
NK 170 FOR Z=1 TO LEN(A$):IF MID
      $(A$,Z,1)=" " THEN FOR Z1
      =1 TO WIDE*HOR:LPRINT:NEX
      T Z1:GOTO 230
JC 180 ST=ST+1:FOR Z1=1 TO WIDE:
      FOR Z2=1 TO HOR:ST=ST+1:I
      F ST>LEN(B$) THEN ST=1
OP 190 PT=ST:FOR Z3=HIGH-1 TO 0
      STEP-1:ASCII=ASC(MID$(TEX
      T$(ASC(MID$(A$,Z,1))-32,Z
      3),Z1,1)):FOR Z4=0 TO 7:I
      F ASCII MOD 2^(Z4+1)>0 TH
      EN ASCII=ASCII-2^Z4:FLAG=
      -1 ELSE FLAG=0
CD 200 FOR Z5=1 TO VERT:PT=PT+1:
      IF PT>LEN(B$) THEN PT=1
PE 210 IF NOT(FLAG) THEN LN$=LN$
      +" " ELSE IF AN=2 THEN LN
      $=LN$+MID$(A$,Z,1) ELSE L
      N$=LN$+MID$(B$,PT,1)
KM 220 NEXT Z5,Z4,Z3:LPRINT TABB
      $;LN$;SPACE$(80-LEN(TABB$
      +LN$));:LN$="":NEXT Z2,Z1
EO 230 NEXT Z:COLOR 7:BEEP:CLS:E
      ND
AE 240 COLOR 7:PRINT"FONT FILES
      NOT FOUND - Insert correc
      t disk and press any key
      to continue":AN$=INPUT$(1
      ):LOCATE 2,1:RESUME 20
```

©

128 Colorswap

Paul W. Carlson

This short machine language routine makes it simple to create dazzling special effects on the Commodore 128 by swapping colors in the multicolor graphics mode. Several BASIC demonstration programs are included to show you how to use the routine. A disk drive is required.

Many different graphics effects are possible in the Commodore 128's multicolor graphics mode. Here is a brief explanation of that mode and the way it is used by "128 Colorswap." A multicolor mode screen consists of 1000 blocks of 32 double-width pixels. Four different color sources can be used within each block of pixels: background, foreground, multicolor 1, and multicolor 2. These color sources correspond to the color source numbers 0, 1, 2, and 3 in the BASIC COLOR statement. Although each of the 1000 blocks of pixels can have its own colors for each of the four color sources, this article explores some of the effects possible when the same four colors are used for the entire screen and three of the four colors are instantly interchanged.

Creating The Machine Language Routine

To begin, type in, save, and run Program 1. This program creates a short machine language (ML) routine on disk using the filename COLORSWAP. You may want to give Program 1 a descriptive name and keep it to create the Colorswap routine on other disks. The other programs demonstrate how you can use the ML routine in your own programs even if you're not an ML programmer.

Demonstration Programs

Program 2 is the first demonstration program. Type it in and save it on the disk containing the ML file COLORSWAP. Now run the program: It displays three colored boxes. Press any key (except Q, which exits the program) to see how the colors are swapped. Each time you press a key, the colors are shifted one box to the right with the rightmost color going into the box on the left. The box on the left is always the current foreground color at the moment a key is pressed. Likewise, the center box is the current multicolor 1 color and the right box is the current multicolor 2 color.

When you press a key, the program calls the ML routine with the statement SYS 2816. This routine replaces the multicolor 2 color with the multicolor 1 color, replaces the multicolor 1 color with the foreground color, and replaces the foreground color with the old multicolor 2 color. If you look at Program 2, you'll see that it executes SYS 2816 once in line 60 before it waits for a keypress in line 70. This is done because the ML routine does not change any colors the first time it is called in a program.

Programs 3, 4, and 5 show how rapid color swapping can simulate movement. Type in and save all of them.

Program 3 creates a red, green, and blue spiral design against a white background. When the pattern is complete, the spiral appears to rotate rapidly. In fact, this illusion is achieved without redrawing anything or swapping screens (page flipping). Instead, the program simply calls the Colorswap routine to swap the colors.

Programs 4 and 5 use a similar technique. Program 4 creates the illusion of rushing through a tunnel. Program 5 has an interesting 3-D effect that's difficult to describe.

Using Colorswap

Colorswap is easy to put to work. Because the ML routine resides in the cassette buffer, it can be BLOADed at any point in your program before the first SYS 2816 that activates it. Keep in mind that no colors are changed the first time you call the ML routine (this is not important, however, if you intend to call the ML many times in succession to simulate animation).

"128 Colorswap" is a machine language utility that makes it possible to create interesting graphics displays. In this screen, different colors in the design are changed rapidly to create an animated, 3-D effect.

A second point to remember is that Colorswap will change the color of *all* non-background pixels on the screen to the current color source colors defined in your BASIC program. A multicolor screen could contain as many as 16 colors, but after you call the ML routine, the number of colors is reduced to four at the most. This might be useful in some applications, but for simulating animation you should create the display using just one color for each of the four color sources.

It's important to time color changes carefully to eliminate flickering in simulated animation. Flickering occurs in cases where the Colorswap routine cannot change the colors everywhere on the screen during the time the raster is outside of the display area. No flashing is visible in Programs 3–5 because the timing is such that the flickering is limited to the top left corner of the screen, where no swapping occurs. The timing in all three demonstration programs is controlled by the same series of statements. (See lines 130–140 of Program 3.) Fortunately, the timing that produces the least amount of flicker also produces a nice rate of color changing. If you want to use a different rate in your own programs, you may have to experiment a bit.

BASIC 7.0 makes it easy to create multicolor graphics screens, and Colorswap can really make those screens come alive. The short demonstration programs in this article just hint at what is possible.

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

Program 1: 128 COLORSWAP File Creator

```
QK 10 T=0:FORN=2816TO2920:READ
      K:T=T+K:POKEN,K:NEXT
JE 20 IFT<>15400THENPRINT"***
      {SPACE}ERROR IN DATA STA
      TEMENTS ***":END
PD 30 BSAVE"COLORSWAP",P2816TO
      P2921
JD 40 PRINT"COLORSWAP SUCCESSF
      ULLY CREATED":END
BP 50 DATA 166,132,164,133,165
      ,134,134,133
DQ 60 DATA 132,134,133,132,10,
      10,10,10
MR 70 DATA 24,101,133,133,250,
      169,0,133
JG 80 DATA 251,133,253,169,216
      ,133,252,169
JF 90 DATA 28,133,254,162,4,16
      0,0,165
DB 100 DATA 134,145,251,165,25
      0,145,253,200
PB 110 DATA 208,245,230,252,23
      0,254,202,208
AM 120 DATA 238,165,1,41,254,1
      33,1,162
KK 130 DATA 4,160,0,132,251,16
      9,216,133
KF 140 DATA 252,165,134,145,25
      1,200,208,249
RB 150 DATA 230,252,202,208,24
      4,169,10,205
KH 160 DATA 18,208,208,251,165
      ,1,41,253
ER 170 DATA 133,1,165,1,9,3,13
      3,1
QM 180 DATA 96
```

Program 2: 128 Colorswap—Demo 1

```
GM 10 BLOAD"COLORSWAP"
MA 20 COLOR0,2:COLOR1,3:COLOR2
      ,6:COLOR3,7:COLOR4,2:GRA
      PHIC3,1
DD 30 X1=10:Y1=70:X2=50:Y2=130
PD 40 FORC=1TO3:BOXC,X1,Y1,X2,
      Y2,,1
EJ 50 X1=X1+50:X2=X2+50:NEXT
MX 60 SYS2816
BA 70 GETKEYA$:IFA$<>"Q"THEN60
JE 80 COLOR0,12:COLOR4,14:GRAP
      HIC0,1:GRAPHICCLR
```

Program 3: 128 Colorswap—Demo 2

```
GM 10 BLOAD"COLORSWAP"
CB 20 COLOR0,2:COLOR1,3:COLOR2
      ,6:COLOR3,7:COLOR4,2
KF 30 GRAPHIC3,1:CX=80:CY=100
DH 40 CIRCLE3,CX,CY,63,90
EG 50 RD=89.5:TP=2*↑:K=9:N=20:
      F=RD/TP:DA=TP/K:DB=TP/N:
      A=0:C=4
AQ 60 FORI=1TOK:B=0:A=A+DA:C=C
      -1:IFC=0THENC=3
FR 70 DRAWC,CX,CY
QH 80 FORJ=1TON:B=B+DB:R=F*B:D
      RAWCTOCX+.7*R*SIN(A+B),C
      Y+R*COS(A+B):NEXTJ,I
QF 90 DRAW3,78,102:A=0
MP 100 FORI=1TOK:A=A+DA:C=C-1:
       IFC=0THENC=3
GQ 110 PAINTC,CX+.65*R*SIN(A),
       CY+.95*R*COS(A),1:NEXT
RK 120 CIRCLE0,CX,CY,63,90
CD 130 FORN=1TO10:NEXT:SYS2816
MS 140 GETA$:IFA$=""THEN130
BD 150 COLOR0,12:COLOR4,14:GRA
       PHIC0,1:GRAPHICCLR
```

Program 4: 128 Colorswap—Demo 3

```
GM 10 BLOAD"COLORSWAP"
CR 20 COLOR0,1:COLOR1,3:COLOR2
      ,6:COLOR3,7:COLOR4,1:GRA
      PHIC3,1
JM 30 C=1:X1=16:X2=144:Y1=10:Y
      2=190
RK 40 FORI=0TO10:XP(I)=X1+1:YP
      (I)=Y1+1
HK 50 BOXC,X1,Y1,X2,Y2
BC 60 C=C-1:IFC=0THENC=3
SM 70 X1=X1+.1*(X2-X1):X2=159-
      X1
AB 80 Y1=Y1+.1*(Y2-Y1):Y2=199-
      Y1:NEXT
QA 90 C=2:FORI=0TO9:C=C-1:IFC=
      0THENC=3
EX 100 PAINTC,XP(I),YP(I),1:NE
       XT
KA 110 GETA$:IFA$<>""THEN130
JS 120 FORN=1TO10:NEXT:SYS2816
       :GOTO110
KB 130 COLOR0,12:COLOR4,14:GRA
       PHIC0,1:GRAPHICCLR
```

Program 5: 128 Colorswap—Demo 4

```
GM 10 BLOAD"COLORSWAP"
MA 20 COLOR0,2:COLOR1,3:COLOR2
      ,6:COLOR3,7:COLOR4,2:GRA
      PHIC3,1
RM 30 CX=80:CY=110:RD=70:TP=2*
      ↑:N=15:F=RD/(2*TP):DB=TP
      /(N+N):C=1
QG 40 FORJ=1TO4.8*N:B=B+DB:R=F
      *B:X=CX+.7*R*SIN(B):Y=CY
      +R*COS(B)
EK 50 IFJ>10THENCIRCLEC,X,Y,.1
      75*R,.25*R:PAINTC,X,Y,0
BC 60 C=C+1:IFC=4THENC=1
RJ 70 NEXT
CF 80 GETA$:IFA$<>""THEN100
CD 90 FORN=1TO10:SYS2816:GOTO8
      0
EX 100 COLOR0,12:COLOR4,14:GRA
       PHIC0,1:GRAPHICCLR
```

©

Six New Operators For Atari BASIC

Rhett Anderson, Assistant Editor

This compact machine language utility adds six useful bitwise operators to Atari BASIC.

Atari BASIC differs from most other BASICs in a number of ways. Although it includes some hardware-related commands (GRAPHICS, STICK, PADDLE, and so on), its lack of bitwise operators makes accessing other hardware features difficult. "Six New Operators for Atari BASIC" adds six bitwise operators to BASIC. The program is published in the form of BASIC statements which you can add to your own programs. Begin your program at line 30.

Bitwise Operators

What are bitwise operators, and what makes them so important? On some computers you may see a line that looks like this:

```
10 POKE 65460,PEEK(65460)
   AND 254
```

This line looks confusing to most Atari programmers because Atari BASIC uses AND only as a *logical operator*. Logical operators consider values to be either true or false. They are often used to create an IF statement that contains two or more logical tests. For instance, this line uses AND as a logical operator:

```
20 IF A=1 AND Y<200 THEN
   GOTO 200
```

In this statement, the computer performs GOTO 200 only when the value of A is 1 *and* the value of Y is less than 200. The AND links together the conditions A=1 and Y<200.

In Atari BASIC, a zero is treated as false and anything else is considered true. Logical operators always return a value of either 0 or 1. Thus, the result of the IF test in line 20 is 0 when one or both conditions are false, and 1 when both of them are true.

A *bitwise operator*, on the other hand, treats each bit of a byte-size value separately. A plain English translation of line 10 would read something like this: "Get the value from memory location 65460 and perform an AND operation with the value 254, treating each bit separately. Store the result back in location 65460." Since 254 is 11111110 in binary, line 10 has the effect of turning off the least significant bit of location 65460 (setting the lowest bit to 0).

Bitwise operators are extremely useful when you need to access one of the Atari's hardware registers (a memory location set aside for controlling a specific hardware feature). Some hardware registers serve more than one purpose, with each bit in the register controlling a different feature. There are many cases where you might want to change the value of just one bit in a hardware register, without disturbing the other bits. That sort of activity is difficult if you don't have bitwise operators.

This program provides a convenient means for performing bitwise operations such as the one in line 10. If you are a bit confused by the preceding explanation, don't lose heart. The last section of this article offers some examples which you can use even if you don't understand binary numbers or bitwise operators fully.

Operator List

The new bitwise operators are XOR (eXclusive OR), BOR (Bitwise OR), BAND (Bitwise AND), BNOT (Bitwise NOT), SHL (SHift Left), and SHR (SHift Right). Let's examine them.

XOR. The result (for each bit) is 1 if one and only one of the operands is 1. So, 1 XOR 1 = 0 and 0 XOR 1 = 1.

BAND. The result is 1 only if both operands are 1. So, 1 AND 1 = 1 and 0 AND 1 = 0.

BOR. The result is 1 if either or both operands are 1. So, 1 OR 1 = 1 and 0 OR 1 = 1.

BNOT. The result is opposite the operand (this operator only accepts one operand).

SHL. Shifts all bits (16 of them) to the left a designated number of times. Each shift is equivalent to a multiplication by 2.

SHR. Shifts all bits to the right a specified number of times. Each shift is equivalent to an integer division by 2.

These operators are accessed with the USR function. Following are examples which show the syntax of each operator.

```
RESULT=USR(XOR,a,b)
RESULT=USR(BAND,a,b)
RESULT=USR(BOR,a,b)
RESULT=USR(BNOT,a)
RESULT=USR(SHL,a,b)
RESULT=USR(SHR,a,b)
```

Each USR statement must include the desired operator (XOR, BAND, and so on) plus two operands (except for BNOT, which takes only one operand). The operands—represented by *a* and *b* in the examples—are the values needed

to perform the operation. The operands may consist of numeric constants or any expressions that evaluate to a numeric value. For instance, both of these lines return the result of 3:

```
10 RESULT=USR(BOR,1,2)
20 A=1:B=(2*A):RESULT=USR
   (BOR,A,B)
```

In each case, the variable RESULT will contain the result of the operation. Of course, you can replace RESULT with any legal Atari BASIC variable name. To save space, the machine language routine includes no error checking, so be sure to include the proper number of parameters. If you don't, you will have to press SYSTEM RESET to regain control of your computer.

Examples

Bitwise operators can be used in many different ways. Following are some examples which you can use in your own programs. Ian Chadwick's book, *Mapping the Atari* (available from COMPUTE! Books), contains much more information about hardware registers and how to use them.

```
B=USR(BAND,A,1):REM B=
1 IF A IS ODD, B=0 IF
A IS EVEN

B=USR(XOR,B,1):REM MAK
ES B=1 IF B WAS 0, MAK
ES B=0 IF B WAS 1.

B=USR(BNOT,B):REM SAME
 AS ABOVE

B=USR(BAND,NUM,255):RE
M RETURNS THE LOW BYTE
 OF NUM

B=USR(SHR,NUM,8):REM R
ETURNS THE HIGH BYTE O
F NUM

A=USR(BAND,8,STICK(0))
:REM RETURNS A 0 IF TH
E JOYSTICK IS PRESSED
RIGHT (AND A 8 IF IT I
SN'T)

A=USR(BAND,1,PEEK(5327
9)):REM RETURNS A 0 IF
 START IS PRESSED, A 1
 IF IT ISN'T.

POKE 623,USR(BOR,PEEK(
623),64):REM ENABLE GT
IA MODE 9. THIS IS INT
ERESTING TO DO IN GRAP
HICS 0.

POKE 562,USR(BAND,PEEK
(562),254):REM TURN OF
F KEYBOARD DEBOUNCE CI
RCUIT.
```

Six New Operators For Atari BASIC

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

```
HD 10 FOR T=1536+128 TO 1775
      :READ A:POKE T,A:NEXT
      T
FN 20 XOR=1536+128:BAND=XOR+
      16:BOR=16+BAND:BNOT=16
      +BOR:SHL=BNOT+12:SHR=S
      HL+13
BA 1010 DATA 32,214,6,165,21
      2,69,214,133
HL 1020 DATA 212,165,213,69,
      215,133,213,96
AN 1030 DATA 32,214,6,165,21
      2,37,214,133
HI 1040 DATA 212,165,213,37,
      215,133,213,96
NK 1050 DATA 32,214,6,165,21
      2,5,214,133
EF 1060 DATA 212,165,213,5,2
      15,133,213,96
HC 1070 DATA 104,104,73,255,
      133,213,104,73
BN 1080 DATA 255,133,212,96,
      32,214,6,166
EI 1090 DATA 214,6,212,38,21
      3,202,208,249
OC 1100 DATA 96,32,214,6,166
      ,214,70,213
KB 1110 DATA 102,212,202,208
      ,249,96,104,133
MI 1120 DATA 216,104,133,217
      ,104,104,133,213
ME 1130 DATA 104,133,212,104
      ,133,215,104,133
FH 1140 DATA 214,165,217,72,
      165,216,72,96
```

Omega Sort

Jonathan J. Holuta

Written entirely in machine language, this fast sorting routine for the Commodore 64 can be used by anyone and does not take away any space from BASIC memory.

If you write programs that handle data, sooner or later you will need a routine to sort items into alphabetical order. There are several sorting methods suitable for use in BASIC, including the bubble sort, shell sort, and quick sort. None of those methods, however, is very efficient for sorting large amounts of data.

"Omega Sort" is a speedy machine language routine which you can use in any BASIC program, even if you don't know a thing about machine language. Program 1 contains the sort routine. Type in this program with the "MLX" machine language entry program found elsewhere in this issue. Here are the starting and ending addresses you'll need when typing in the program:

Starting address: C000
Ending address: C377

Don't forget to save a copy of the program after you finish typing it in. If you want use Program 2 to test the sorting routine, save the data from Program 1 with the name OMEGASORT, since that's the name Program 2 looks for.

Omega Sort can sort 1000 randomly ordered strings in alphabetical order in less than six seconds. To see the routine at work, type in and save Program 2, the BASIC demonstration program. If you are using tape instead of disk, change the 8 to a 1 in line 10 of Program 2.

When you run Program 2, it loads the machine language routine from disk or tape into memory. Then it prompts you to enter the number of strings you wish to sort. To create 1000 random strings, for instance, type 1000 and press RETURN. The program prints all of the strings on the screen in their original order, then it sorts them alphabetically. When the sorting is done, the program displays the strings in the new order, one screenful at a time. Press any key to view the next page of data, or press f1 to exit the program.

How To Use It

To use Omega Sort, your program must begin by loading the machine language routine into memory. The first line of Program 2 demonstrates how this is done.

Some machine language sorting routines sort only one dimension of a multidimensional array, which is not always convenient. To demonstrate why, suppose that you have an address file program that stores a list of names and addresses in a two-dimensional array as shown here:

```
N$(1,1)=name 1
N$(1,2)=street 1
N$(1,3)=city 1
N$(1,4)=state 1
N$(1,5)=zip code 1
N$(1,6)=phone 1
```

Each full entry contains six separate items: the name, street, city, state, zip code, and phone number. In a real program, of course, you might have dozens or even hundreds of such entries. The name for entry 2 would be contained in N$(2,1), and so forth.

If you sort the first dimension of this array (name), then the names will be mismatched with the other data items. The name for entry number 1 might be matched with the street for entry 36, and so on.

Instead of sorting the strings themselves, Omega Sort sorts a numeric index array. Each element of the numeric array points to one data set in the string array. The advantage of this method is that all the items within each data set remain in their original order. In addition to great speed, this gives you more flexibility in using string arrays.

In Program 2, the string array is named A$, and the index array is named N%. Note that the index array must be an integer array (one whose name ends with %). Any legal Commodore variable names may be used, provided you follow this simple rule.

Calling The Machine Language

Like other machine language routines, Omega Sort is called with a SYS command. In addition to the command itself, which includes the starting address of the machine code, you must supply three items of information: the number of elements to sort, the name of the string array, and the name of the index array. Here is an example:

```
100 SYS 49152,N,N$(0),N%(0)
```

In this statement, the variable *N* indicates the number of elements to be sorted, and the variable *N$(0)* indicates the name of the array you wish to sort. If there are 40 elements in the N$ array, for instance,

you would set N to 40 before executing line 100. Or, you could just replace N with the number 40. The variable *N%* is the index array.

Once the sorting is complete, the index array contains the new order. To gain access to the sorted data, you must refer to elements of the string array through the index. Look at line 110 of Program 2. The expression A$(A%(X)) causes PRINT to display the elements of A$ in the order contained in the A% array. Remember, Omega Sort rearranges the order of the numeric index array, not the string array itself. Each element of the index array points to one element of the string array.

The SYS statement for a multidimensional array is the same, except that you must specify which dimension to sort. Here is an example:

```
100 SYS 49152,N,N$(0,3),N%(0)
```

For the address array mentioned above, the preceding statement would sort the addresses according to the array's third dimension (city). This statement would sort it according to the first dimension (name):

```
100 SYS 49152,N,N$(0,1),N%(0)
```

This statement would sort the address array by its fifth element (zip code):

```
100 SYS 49152,N,N$(0,5),N%(0)
```

Here is an example line that would print the elements of the address array in their new order:

```
110 FOR X=0 TO N:PRINT X,N$(N%(X)
    ,5):NEXT
```

You can use this routine without knowing how it works, but, for those who are interested, here is a brief explanation. Omega Sort first stores important zero page pointers in the cassette buffer so it can use these locations for its own purposes. Then it determines where in memory the arrays reside. In the case of strings, the actual text is stored from the top of BASIC memory in a downward direction. The array storage space (located just above the end of BASIC program text) contains a series of pointers to the strings in high memory. Omega Sort checks the pointers and then changes the values of the integer array to match the alphabetical order of the strings themselves. When finished, it restores the contents of the zero page and returns to BASIC. The entire process works so quickly that it can sort a hundred strings in less than a second.

Program 1: Omega Sort

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

```
C000:20 8E C0 20 60 C0 20 23 B8
C008:C0 E6 D9 D0 02 E6 DA 38 85
C010:A5 DD E9 02 85 DD A5 DE 07
C018:E9 00 85 DE 20 48 C1 20 F2
C020:9A C0 60 A5 DD 85 DF A5 EF
C028:DE 85 E0 A9 00 85 E1 85 90
C030:E2 A0 01 A5 E1 91 DF 88 63
C038:A5 E2 91 DF C5 DA F0 16 07
C040:18 A5 DF 69 02 85 DF A5 55
C048:E0 69 00 85 E0 E6 E1 D0 24
C050:E0 E6 E2 4C 31 C0 A5 E1 D6
C058:C5 D9 F0 03 4C 40 C0 60 C6
C060:20 FD AE 20 9E AD 20 F7 2D
C068:B7 A5 14 85 D9 A5 15 85 1F
C070:DA 20 FD AE 20 9E AD A5 8E
C078:47 85 DB A5 48 85 DC 20 07
C080:FD AE 20 9E AD A5 47 85 B2
C088:DD A5 48 85 DE 60 A0 19 96
C090:B9 D8 00 99 3C 03 88 D0 8E
C098:F7 60 A0 19 B9 3C 03 99 32
C0A0:D8 00 88 D0 F7 60 A5 DD 17
C0A8:85 DF A5 DE 85 E0 A0 02 7A
C0B0:18 A5 DF 6D 72 C3 85 DF 08
C0B8:A5 E0 6D 73 C3 85 E0 88 A8
C0C0:D0 EE 60 A5 DD 85 E1 A5 3B
C0C8:DE 85 E2 A0 02 18 A5 E1 1F
C0D0:6D 74 C3 85 E1 A5 E2 6D CF
C0D8:75 C3 85 E2 88 D0 EE 60 AA
C0E0:A0 01 B1 DF 8D 76 C3 88 7D
C0E8:B1 DF 8D 77 C3 A5 DB 85 56
C0F0:E6 A5 DC 85 E7 A0 03 18 23
C0F8:A5 E6 6D 76 C3 85 E6 A5 C3
C100:E7 6D 77 C3 85 E7 88 D0 AB
C108:EE A0 02 B1 E6 99 EA 00 F9
C110:88 10 F8 60 A0 01 B1 E1 4F
C118:8D 76 C3 88 B1 E1 8D 77 A8
C120:C3 A5 DB 85 E8 A5 DC 85 DF
C128:E9 A0 03 18 A5 E8 6D 76 CC
C130:C3 85 E8 A5 E9 6D 77 C3 26
C138:85 E9 88 D0 EE A0 02 B1 C6
C140:E8 99 EA 00 88 10 F8 60 D2
C148:A2 01 A9 01 9D 80 C3 A9 C2
C150:00 9D AE C3 E8 A5 D9 9D 7C
C158:80 C3 A5 DA 9D AE C3 A9 48
C160:02 8D 7A C3 A9 00 8D 7B B7
C168:C3 AE 7A C3 BD 80 C3 8D 0A
C170:78 C3 BD AE C3 8D 79 C3 CE
C178:AD 7A C3 D0 03 CE 7B C3 05
C180:CE 7A C3 AE 7A C3 BD 80 4C
C188:C3 8D 7C C3 BD AE C3 8D DA
C190:7D C3 AD 7A C3 D0 03 CE 57
C198:7B C3 CE 7A C3 AD 7C C3 DD
C1A0:8D 72 C3 AD 7D C3 8D 73 64
C1A8:C3 AD 78 C3 8D 74 C3 AD 38
C1B0:79 C3 8D 75 C3 18 AD 7C 41
C1B8:C3 6D 78 C3 8D 7E C3 AD 60
C1C0:7D C3 6D 79 C3 8D 7F C3 50
C1C8:6E 7F C3 6E 7E C3 A5 DD EE
C1D0:85 E3 A5 DE 85 E4 A0 02 B5
C1D8:18 A5 E3 6D 7E C3 85 E3 17
C1E0:A5 E4 6D 7F C3 85 E4 88 9C
C1E8:D0 EE A0 00 B1 E3 8D 77 54
C1F0:C3 C8 B1 E3 8D 76 C3 A5 70
C1F8:DB 85 F0 A5 DC 85 F1 A0 C5
C200:03 18 A5 F0 6D 76 C3 85 23
C208:F0 A5 F1 6D 77 C3 85 F1 4C
C210:88 D0 EE A0 02 B1 F0 99 48
C218:ED 00 88 10 F8 20 A6 C0 FC
C220:20 E0 C0 A0 FF C8 C4 ED AA
C228:B0 0F C4 EA B0 14 B1 EB 36
C230:D1 EE 90 0E F0 EF 4C 5A 87
C238:C2 A5 EA C5 ED 90 03 4C 46
C240:5A C2 EE 72 C3 D0 03 EE FE
C248:73 C3 18 A5 DF 69 02 85 04
C250:DF A5 E0 69 00 85 E0 4C 06
C258:1D C2 20 C3 C0 20 14 C1 CD
C260:A0 FF C8 C4 EA B0 0F C4 98
C268:ED B0 14 B1 EE D1 EB 90 D5
C270:0E F0 EF 4C 9A C2 A5 ED 15
C278:C5 EA 90 03 4C 9A C2 AD DD
C280:74 C3 D0 03 CE 75 C3 CE 1E
C288:74 C3 38 A5 E1 E9 02 85 DA
C290:E1 A5 E2 E9 00 85 E2 4C 93
C298:5A C2 AD 73 C3 CD 75 C3 EC
C2A0:90 12 F0 03 4C E2 C2 AD 62
C2A8:72 C3 CD 74 C3 90 05 F0 B4
C2B0:03 4C E2 C2 A0 00 B1 DF 9B
C2B8:48 C8 B1 DF 48 B1 E1 91 27
C2C0:DF 88 B1 E1 91 DF A0 01 FA
C2C8:68 91 E1 88 68 91 E1 EE E7
C2D0:72 C3 D0 03 EE 73 C3 AD 45
C2D8:74 C3 D0 03 CE 75 C3 CE 76
C2E0:74 C3 AD 73 C3 CD 75 C3 82
C2E8:90 12 F0 03 4C FF C2 AD 1F
C2F0:72 C3 CD 74 C3 90 05 F0 FC
C2F8:03 4C FF C2 4C 1D C2 AD 49
C300:7D C3 CD 75 C3 90 10 F0 B9
C308:03 4C 43 C3 AD 7C C3 CD 7D
C310:74 C3 90 03 4C 43 C3 EE EA
C318:7A C3 D0 03 EE 7B C3 AE B3
C320:7A C3 AD 7C C3 9D 80 C3 AC
C328:AD 7D C3 9D AE C3 EE 7A 15
C330:C3 D0 03 EE 7B C3 E8 AD 87
C338:74 C3 9D 80 C3 AD 75 C3 2A
C340:9D AE C3 AD 72 C3 8D 7C CF
C348:C3 AD 73 C3 8D 7D C3 CD 7F
C350:79 C3 90 03 4C 5A C3 4C 67
C358:A9 C1 D0 08 AD 7C C3 CD 74
C360:78 C3 90 F3 AD 7B C3 D0 1A
C368:06 AD 7A C3 D0 01 60 4C 81
C370:69 C1 EA 00 00 00 00 00 7A
```

Program 2: BASIC Demonstration

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

```
DG 10 IF Z=0 THEN Z=1:LOAD"OME
      GASORT",8,1
QQ 20 POKE 53281,1:PRINT"{CLR}
      {2 DOWN}{BLK}"
MH 30 DEFFNA(X)=INT(X*100+.5)/
      100
MC 40 INPUT"HOW MANY";N:DIMA$(
      N),A%(N)
PA 50 A$="ABCDEFGHIJKLMNOPQRST
      UVWXYZ"
QQ 60 FORX=0TON:A$(X)=MID$(A$,
      RND(1)*18+1,RND(1)*5+3):
      PRINTX,A%(X),A$(X)
RJ 70 NEXT
FK 80 PRINT"{DOWN}{RVS}SORTING
      ":T1=TI
GD 90 SYS 49152,N,A$(0),A%(0)
EG 100 T2=TI:TM=(T2-T1)/60:PRI
       NT"{CLR}{DOWN}"
HF 110 FORX=0TON:PRINTX,A%(X),
       A$(A%(X))
BG 120 IFPEEK(214)<21THEN170
BR 130 PRINT"{DOWN}HIT ANY KEY
        TO CONTINUE:{2 SPACES}
       F1 TO END
BQ 140 GETB$:IFB$=""THEN140
CH 150 IFB$="{F1}"THENX=N-18
GH 160 PRINT"{CLR}{DOWN}"
SQ 170 NEXT
FE 180 PRINT"{2 DOWN}{RVS}"FNA
       (TM)"{OFF}SECONDS"
```

Atari Disk Sector Editor

Marcelo Adapon

With this utility you can view and change the contents of any sector on a standard floppy disk. The program works with Atari DOS 2.0 and 2.5 and runs on any Atari 800XL, 65XE, or 130XE computer. (The program will not work on the older 400 or 800 models.) A disk drive and joystick are required. Recommended for intermediate and advanced programmers.

If you are interested in learning about Atari disk organization, or if you have ever needed to recover an accidentally deleted disk file, "Atari Disk Sector Editor" can be a very useful tool. It's a convenient, menu-driven utility which allows you to display the contents of any disk sector on the screen and modify any byte or series of bytes within the sector. (A disk editor is a very powerful tool—if misused, it can easily scramble an entire disk, destroying its contents forever. To avoid losing important data, you should *practice using this program on an unimportant disk* until you are familiar with its use.)

Type in the program and save it to disk. Notice line 5: To edit an enhanced-density DOS 2.5 disk, you'll need to change the DENSITY=0 in that line to DENSITY=1. The program uses several of the less common screen editor sequences, so be sure to refer to the "Guide to Typing In Programs" article elsewhere in this issue if you see something in braces ({ }) that you don't understand. For example, the {5 DEL LINE} in line 470 means to type the delete-line sequence, ESC–SHIFT–DELETE, five times.

When you run Disk Sector Editor, it spends a few moments installing machine language subroutines; then it displays the menu screen. This screen lists all the commands available in the program. The menu disappears when you display a disk sector. Use the joystick to move the cursor from one byte to another in the sector display. You can go back to the menu at any time by pressing the question mark key (?).

Command List

Here is a complete list of the program's commands:

R. Reads the sector indicated by the number in the sector indicator and displays its contents on the screen.

W. Writes the current sector back to disk, including any changes you have made while editing the sector.

C. Changes to a new sector.

T. Activates text input mode. Text mode lets you change the contents of the byte under the cursor by typing a key. (Don't type too quickly—input is rather slow in this mode.) Exit this mode by pressing CTRL-CLR.

H. Activates hexadecimal input mode. As with text mode, this mode lets you change the contents of a byte. However, the new value is typed as a hexadecimal value. For instance, typing the characters AA changes the contents of the byte under the cursor to hexadecimal $AA. Exit hexadecimal mode by typing ZZ.

D. Activates decimal input mode. This mode works the same as text and hexadecimal mode, except that entries are in decimal. Exit by typing −1.

L. Displays sector link information. This function shows the data contained in the last three bytes of the current sector. These bytes show the number of active bytes in the sector, the file number, and the next sector in the chain of linked sectors. Note that if the last byte is zero, you have reached the final sector in the chain (the end of the file).

S. Shows the decimal value of the byte under the cursor.

A. Shows the character in ATASCII and internal format.

N. The Next command automatically reads the next sector in the file chain and increments the sector indicator to that sector number.

+. Pressing the plus key (+) causes the program to read the next sector in numerical order. If you execute this command from sector 720 (standard density) or 1010 (enhanced density), the program proceeds to sector 1.

−. Pressing the minus key (−) causes the program to read the previous sector in numerical order. If you execute this command from sector 1, the program backs up to sector 720 (standard density) or 1010 (enhanced density).

?. The question mark key (?) returns you to the main menu, which lists all the program's commands.

Among other things, this program allows you to recover a file that was deleted accidentally. Before you try to recover an actual file, it's a good idea to practice this process with a dummy file on an unimportant disk. For instance, create a dummy file by saving a one-line BASIC program to disk; then delete the file to set up the conditions for recovering it. After you know that you can successfully recover the dummy file, you can proceed to restore important files.

Directory Records

To begin the recovery process, read the directory sectors (361–368) to find out whether the filename of the deleted file still exists. It's important to understand the format of file records within the disk directory. Each record contains 16 bytes, whose significance is explained as follows.

Byte 0: Status

The status byte records the file's status, which is one of four possible values:

$40 = normal
$43 = unclosed
$80 = deleted
$20 = locked

The status byte for a deleted file appears on the screen as the heart character.

Bytes 1–2: Length

These bytes show the length of the file in low-byte/high-byte format. To convert from low-byte/high-byte format to a decimal number, use the BASIC statement PRINT LO + 256 * HI, where LO equals the low-byte value and HI equals the high byte.

Bytes 3–4: Starting sector

This pair of bytes indicates the sector where the file begins. This value is also in low-byte/high-byte format.

Bytes 5–12: Filename

The first part of the filename (the eight characters before the period) is contained in these bytes.

Bytes 13–15: Extension

These three bytes contain the three-character extension which appears after the period in a filename.

When you view a file record with this program, each record takes up two lines of the display. Each record starts on a line that ends with a zero (10, 20, and so on). To recover a deleted file, you need only change that file's status byte from $80, meaning that it's deleted, to $40, the normal file type. Once this is done, write the sector back to disk.

File Recovery

The best time to recover a file is immediately after it has been deleted, before any other files have been created or updated. That way, you can be reasonably certain that no part of the deleted file has been overwritten by another file. After recovering a file, you should exit the program and attempt to read the file normally, to make sure all of it is present. (Don't write to that file or any other file on the disk, however, or you may destroy your chances of recovering it.)

With the file intact, only one job remains. You have changed the file's status back to normal, but you must still update the disk's VTOC (Volume Table Of Contents) so that DOS knows the file's sectors are in use again. Copy the recovered file to a second disk; then insert the original disk and delete the file again from the DOS menu. Now copy the file back from the second disk to the original. DOS updates the VTOC and the file is restored completely.

Recovery is much more difficult in cases where the deleted file has no entry in the directory sectors or where part of it is missing after you've restored it to normal status. Since the directory holds no clue as to the file's length or location, you have to look through every sector on the disk to find the beginning of the file, then determine its length manually by chaining through all its sectors until you reach the final sector. Once that has been done, you have to construct a new file record in the directory and update the VTOC as well. It's possible to recover a file in this way, but only if none of it has been overwritten by other files. And this method depends on your ability to recognize the file's first sector amongst all the other sectors on the disk. Unless the file is absolutely irreplaceable, you may find it more time-efficient to recreate the file by using the program that created it in the first place.

Atari Disk Sector Editor

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

```
FE 5  DENSITY=0:REM DENSITY=1
      IF USING DOS 2.5 ENHAN
      CED DENSITY
MA 10 IF PEEK(1536)<>173 THE
      N GOSUB 1150
FF 20 P2=PEEK(106)-5:POKE 10
      6,P2:GRAPHICS 0:SC1=PE
      EK(88):SC2=PEEK(89):SC
      RN=P2*256:DL=PEEK(560)
      +256*PEEK(561)
AF 30 POKE 752,1:SECTOR=1
CN 40 DIM R$(1),BUF$(128),CM
      D$(1):DRIVE=1:BUF$=CHR
      $(0):BUF$(128)=BUF$:BU
      F$(2)=BUF$:ADDR=ADR(BU
      F$)
JA 50 DIM HX$(16),HXN$(3),HX
      N1$(2):HX$="0123456789
      ABCDEF"
ID 60 DIM ML1$(26),ML2$(19):
      RESTORE 62
FI 62 FOR I=1 TO 26:READ BYT
      :ML1$(I)=CHR$(BYT):NEX
      T I
FN 64 FOR I=1 TO 19:READ BYT
      :ML2$(I)=CHR$(BYT):NEX
      T I
NP 66 DATA 104,104,133,1,104
      ,133,0,162,0,160,0,169
      ,0,145,0,200,208,249,2
      30,1,232,224,4,208,242
      ,96
HD 68 DATA 72,138,72,162,0,1
      69,0,157,192,158,232,2
      24,40,208,246,104,170,
      104,64
FC 70 A=USR(ADR(ML1$),SCRN):
      A=ADR(ML2$):POKE A+9,I
      NT((SCRN+960)/256):POK
      E A+8,SCRN+960-PEEK(A+
      9)*256
JF 80 POKE DL+10,130:POKE 51
      3,INT(A/256):POKE 512,
      A-PEEK(513)*256:POKE 5
      4286,192
FH 90 ? "R - READ SECTOR "
PD 100 ? "W - WRITE SECTOR"
MB 110 ? "C - CHANGE SECTOR
      READ/WRITE NUM"
HE 120 ? "T - ENTER TEXT DAT
      A"
AJ 130 ? "H - ENTER HEX DATA
      "
BA 140 ? "D - ENTER DECIMAL
      DATA"
EJ 150 ? "L - PRINT SECTOR L
      INK INFO"
EH 160 ? "S - SHOW DECIMAL V
      ALUE"
JM 170 ? "A - CHARACTER REPR
      ESENTATIONS"
KA 180 ? "N - NEXT SECTOR IN
       CHAIN"
JK 190 ? "+ - DISPLAY NEXT S
      ECTOR"
NC 200 ? "- - DISPLAY PREVIO
      US SECTOR"
GL 210 ? "? - HELP SCREEN"
CM 220 ? " USE JOYSTICK TO M
      OVE CURSOR "
NH 230 ? " PRESS A KEY TO CO
      NTINUE "
MN 240 OPEN #1,4,0,"K:"
IE 250 GET #1,B:IF FL<>1 THE
      N GOSUB 600
FD 260 FL=1:GOSUB 650:POKE 7
      54,1:POKE 694,0:POKE
      702,64
LB 270 IF PEEK(754)<>1 THEN
      450
KE 280 D=PEEK(632):IF D=15 T
      HEN 270
DH 290 POKE LOC,PEEK(LOC)-12
      8:POKE LOC+1,PEEK(LOC
      +1)-128
BA 300 POSITION 28,Y+1:? " "
JA 310 POKE TEMP,PEEK(TEMP)-
      128:POKE TEMP+1,PEEK(
      TEMP+1)-128
BH 320 POSITION 6+X*3,0:? X:
      POSITION 29+X,0:? X
GC 330 IF D=10 OR D=14 OR D=
      6 THEN Y=Y-1
DH 340 IF D=9 OR D=13 OR D=5
       THEN Y=Y+1
```

```
GC 350  IF D=10 OR D=11 OR D=
        9 THEN X=X-1
AH 360  IF D=6 OR D=7 OR D=5
        THEN X=X+1
OK 370  IF X>7 THEN X=0
OJ 380  IF X<0 THEN X=7
BN 390  IF Y>15 THEN Y=0
BD 400  IF Y<0 THEN Y=15
CP 410  LOC=SCRN+45+40*Y+X*3:
        POKE LOC,PEEK(LOC)+12
        8:POKE LOC+1,PEEK(LOC
        +1)+128:POSITION 28,Y
        +1:? "{ESC}{RIGHT}"
BE 420  TEMP=SCRN+42+40*Y:POK
        E TEMP,PEEK(TEMP)+128
        :POKE TEMP+1,PEEK(TEM
        P+1)+128
EP 430  POSITION 6+X*3,0:? CH
        R$(176+X):POSITION 29
        +X,0:? CHR$(176+X)
GK 440  GOTO 270
CC 450  GET #1,B:R$=CHR$(B):P
        OKE 754,1
LE 460  IF R$="?" THEN GOSUB
        660:GOTO 250
IK 470  IF R$="R" THEN CMD$="
        R":GOSUB 1270:GOSUB 6
        00:POSITION 0,18:? "
        {5 DEL LINE}":GOTO 27
        0
JF 480  IF R$="W" THEN CMD$="
        W":GOSUB 1270:GOSUB 6
        00:POSITION 0,18:? "
        {5 DEL LINE}":GOTO 27
        0
DC 490  IF R$="C" THEN 670
DC 500  IF R$="H" THEN 880
CL 510  IF R$="D" THEN 840
GB 520  IF R$="T" THEN 1000
DE 530  IF R$="L" THEN 740
BI 540  IF R$="+" THEN 690
BE 550  IF R$="-" THEN 710
DN 560  IF R$="S" THEN 730
DA 570  IF R$="A" THEN 770
DK 580  IF R$="N" THEN 820
HA 590  GOTO 270
MP 600  GOSUB 650:X=0:Y=0:LOC
        =SCRN+45
MO 610  POSITION 0,0:? "LINE
          1  2  3  4  5  6
         7 1234567":A=USR(15
        36,ADDR)
ON 620  ? :? "SECTOR # TO BE
        WRITTEN/READ:";SECTOR
        ;"  "
HM 630  TEMP=LOC-3:POKE TEMP,
        PEEK(TEMP)+128:POKE T
        EMP+1,PEEK(TEMP+1)+12
        8
LJ 640  POKE LOC,PEEK(LOC)+12
        8:POKE LOC+1,PEEK(LOC
        +1)+128:POSITION 28,1
        :? "{ESC}{RIGHT}":RET
        URN
PD 650  POKE DL+4,0:POKE DL+5
        ,P2:POKE 88,0:POKE 89
        ,P2:RETURN
KO 660  POKE DL+4,SC1:POKE DL
        +5,SC2:POKE 88,SC1:PO
        KE 89,SC2:RETURN
IK 670  POSITION 0,18:? "
        {5 DEL LINE}SECTOR NU
        MBER";:TRAP 670:INPUT
         SECTOR:POSITION 30,1
        7:? SECTOR;"  "
DN 680  POSITION 0,18:? "
        {DEL LINE}":GOTO 270
NO 690  SECTOR=SECTOR+1:IF SE
        CTOR>1010 THEN SECTOR
        =1
KD 691  IF DENSITY=0 AND SECT
        OR>720 THEN SECTOR=1
DD 700  R$="R":GOTO 470
NH 710  SECTOR=SECTOR-1:IF SE
        CTOR<1 THEN SECTOR=10
        10
AE 711  IF DENSITY=0 AND SECT
        OR>720 THEN SECTOR=72
        0
PA 720  R$="R":GOTO 470
AF 730  POSITION 0,18:? "
        {5 DEL LINE}THE DECIM
        AL VALUE OF BYTE #";Y
        *8+X:? "IS EQUAL TO "
        ;PEEK(ADDR+8*Y+X):GOT
        O 280
AK 740  FN=INT(PEEK(ADDR+125)
        /4):NSEC=PEEK(ADDR+12
        6)+256*(PEEK(ADDR+125
        )-FN*4)
EJ 750  POSITION 0,18:? "
        {5 DEL LINE}DOS FILE
        NUMBER ";FN:? "NEXT S
        ECTOR IN THIS FILE IS
         ";NSEC:AC=PEEK(ADDR+
        127)
FJ 760  ? "THERE ARE ";AC;" A
        CTIVE BYTES":? "IN TH
        IS SECTOR":GOTO 270
OC 770  POSITION 0,18:? "
        {5 DEL LINE}ASCII
        {10 SPACES}INTERNAL":
        ? "GRAPHIC ASC
        {4 SPACES}GRAPHIC INT
        "
EO 780  V=PEEK(Y*8+X+ADDR):IF
         V=155 THEN V=27
AK 790  POSITION 10,18:? CHR$
        (27);CHR$(V):POKE SCR
        N+750,V
EM 800  V=V-INT(V/64)*64+64:P
        OSITION 15,19:? CHR$(
        27);CHR$(V):POKE SCRN
        +790,V+64-(V+64>128)*
        128
GL 810  GOTO 270
GI 820  FN=INT(PEEK(ADDR+125)
        /4):NSEC=PEEK(ADDR+12
        6)+256*(PEEK(ADDR+125
        )-FN*4):IF NSEC=0 THE
        N GOTO 260
GH 830  SECTOR=NSEC:CMD$="R":
        R$="R":GOTO 470
LM 840  POSITION 0,18:? "
        {5 DEL LINE}TYPE IN D
        ECIMAL NUMBER THEN RE
        TURN":? "TYPE -1 TO E
        ND DECIMAL ENTRY MODE
        "
EP 850  POSITION 2,20:? "
        {DEL LINE}NUMBER TO R
        EPLACE ";PEEK(ADDR+Y*
        8+X);" :";:TRAP 850:I
        NPUT V:TRAP 40000
FL 860  IF V=-1 THEN POSITION
         0,18:? "{5 DEL LINE}
        ":GOTO 260
PG 870  GOSUB 1060:GOTO 850
JL 880  POSITION 0,18:? "
        {5 DEL LINE}TYPE IN H
        EXADECIMAL NUMBERS AN
        D RETURN";
MH 890  ? "TYPE IN ZZ TO END
        HEX ENTRY MODE"
GO 900  V=PEEK(ADDR+Y*8+X):HX
        =V:GOSUB 1260:POSITIO
        N 0,20:? "{DEL LINE}H
        EX NUMBER TO REPLACE
        $";HXN$;" :?":INPUT H
        XN$
BG 910  TRAP 900:HXN1$="00":H
        XN1$(3-LEN(HXN$),2)=H
        XN$:HXN$=HXN1$:TRAP 4
        0000
KO 920  IF HXN$="ZZ" THEN POS
        ITION 0,18:? "
        {5 DEL LINE}":GOTO 26
        0
DD 930  H=ASC(HXN$(1,1)):IF (
        H<48 OR H>57) AND (H<
        65 OR H>70) THEN ? "
        {BELL}":GOTO 900
EK 940  L=ASC(HXN$(2,2)):IF (
        L<48 OR L>57) AND (L<
        65 OR L>70) THEN ? "
        {BELL}":GOTO 900
ME 950  IF H<65 THEN H=H-48:G
        OTO 970
AD 960  H=H-55
NE 970  IF L<65 THEN L=L-48:G
        OTO 990
AN 980  L=L-55
BC 990  V=H*16+L:GOSUB 1060:G
        OTO 900
HB 1000  POSITION 0,18:? "
         {5 DEL LINE}ASCII CH
         ARACTER ENTRY MODE"
IC 1010  POSITION 2,19:? "PRE
         SS CTRL-CLR TO END T
         EXT ENTRY MODE"
DM 1020  V=PEEK(ADDR+Y*8+X):P
         OSITION 0,20:? "
         {DEL LINE}CHARACTER
         TO REPLACE ";:IF V=1
         55 THEN ? "RETURN":G
         OTO 1020
KB 1030  ? CHR$(27);CHR$(V)
LC 1040  GET #1,V:IF V=158 TH
         EN POSITION 0,18:? "
         {5 DEL LINE}":GOTO 2
         60
ED 1050  GOSUB 1060:GOTO 1020
PG 1060  POSITION X*3+5,Y+1:H
         X=V:GOSUB 1260:? HXN
         $:POSITION X+29,Y+1:
         ? CHR$(27);:IF V=155
          THEN ? CHR$(27):GOT
         O 1080
KP 1070  ? CHR$(V)
EL 1080  POSITION 6+X*3,0:? X
         :POSITION 29+X,0:? X
PP 1090  POKE ADDR+Y*8+X,V
MH 1100  X=X+1:IF X>7 THEN X=
         0:POSITION 28,Y+1:?
         " ";:Y=Y+1:IF Y>15 T
         HEN Y=0
LP 1110  POKE TEMP,PEEK(TEMP)
         -128:POKE TEMP+1,PEE
         K(TEMP+1)-128
EC 1120  TEMP=SCRN+42+40*Y:PO
         KE TEMP,PEEK(TEMP)+1
         28:POKE TEMP+1,PEEK(
         TEMP+1)+128
FP 1130  LOC=SCRN+45+40*Y+X*3
         :POKE LOC,PEEK(LOC)+
         128:POKE LOC+1,PEEK(
         LOC+1)+128:POSITION
         28,Y+1:? "{ESC}
         {RIGHT}"
JI 1140  POSITION 6+X*3,0:? C
         HR$(176+X):POSITION
         29+X,0:? CHR$(176+X)
         :RETURN
DM 1150  RESTORE 1160:FOR A=0
          TO 207:READ B:POKE
         A+1536,B:NEXT A:RETU
         RN
AJ 1160  DATA 173,6,228,141,1
         89,6,238,189,6,173,7
         ,228,141,190,6,169,0
         ,141,253,6,141,254
DB 1170  DATA 6,141,255,6,165
         ,10,141,208,6,165,11
         ,141,209,6,104,104,1
         33,11,104,133,10,174
```