

COMPUTE!

\$3.00
June
1987
Issue 85
Vol. 9, No. 6
\$4.25 Canada
02193
ISSN 0194-357X

The Leading Magazine Of Home, Educational, And Recreational Computing

Laser Chess™

A fascinating, futuristic two-player strategy game—our best ever. For the Commodore 64, Apple II, Amiga, and Atari XL/XE.

64 RAMdisk

Access data instantly with this "disk drive in your computer"

Full-Screen Editor For Applesoft

Easy editing with this feature-packed programmer's aide

Fast Fractal Landscapes For IBM

Atari XL/XE Super Editor

Add four powerful new editing features

Super Hi-Res Graphics And Sound On The Apple IIGs

Amiga Disk-Based Character Sets

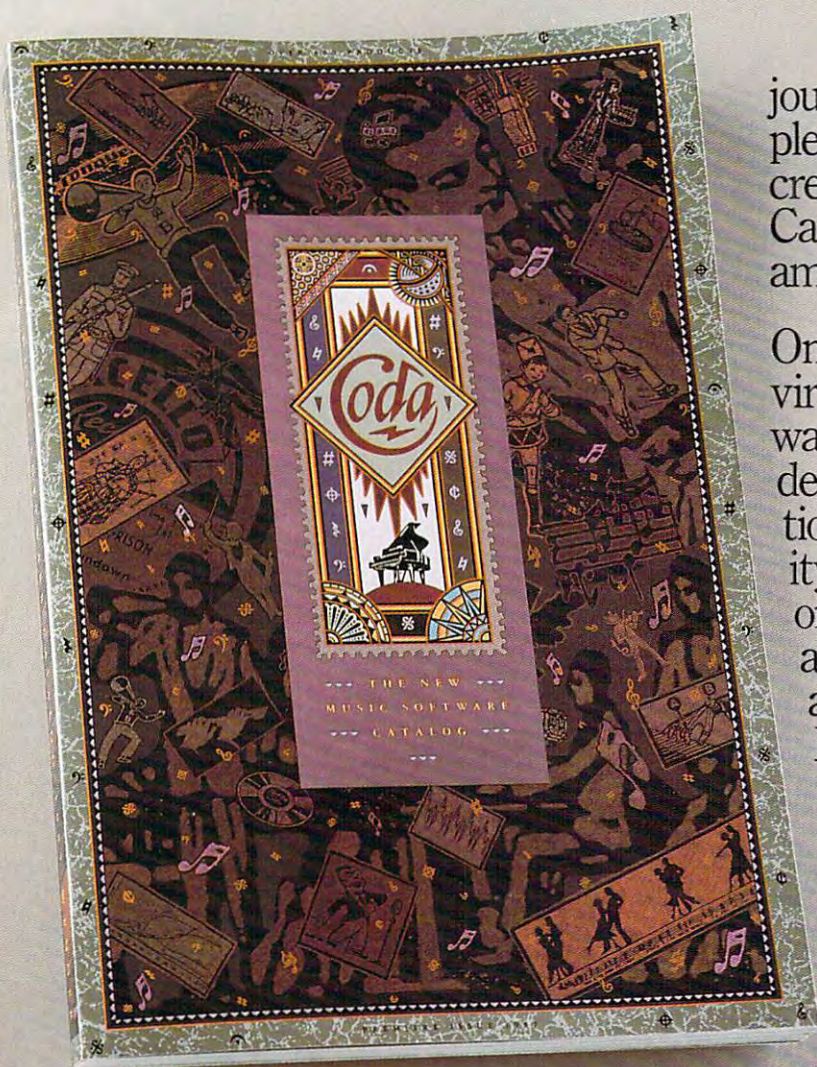
College Fund Planner For The IBM PC

How to calculate the funds needed for your child's education



71486 02193 3

How To Find Singing Frogs, Bathtub Reverb, Secrets Of Analog And Other Information Beethoven Would Have Killed For.



You're about to embark on a journey through the most complete music software catalog ever created. It's called the Coda Catalog. 160 pages of intrigue, amusement, and information.

On over 600 products, including virtually every piece of music software that exists today. Coda is detailed with whimsical illustrations. And written with a simplicity you'll appreciate. Use it to order software, books, videos, and equipment. All at the guaranteed lowest price. For Apple, IBM, Macintosh, Amiga, Atari, and Commodore computers.

Quite simply, Coda is the best source of music software in the world. Or as one critic so eloquently put it, "Beethoven would have killed for this

information." Only \$4. Order by calling toll free 1-800-843-1337. Or collect 612-854-9554. Oh, by the way, the singing frog is on page 114.

Wenger Corporation, Music Learning Division, 1401 East 79th Street, Minneapolis, MN 55420-1590

©1987, Wenger Corp.



www.commodore.ca



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

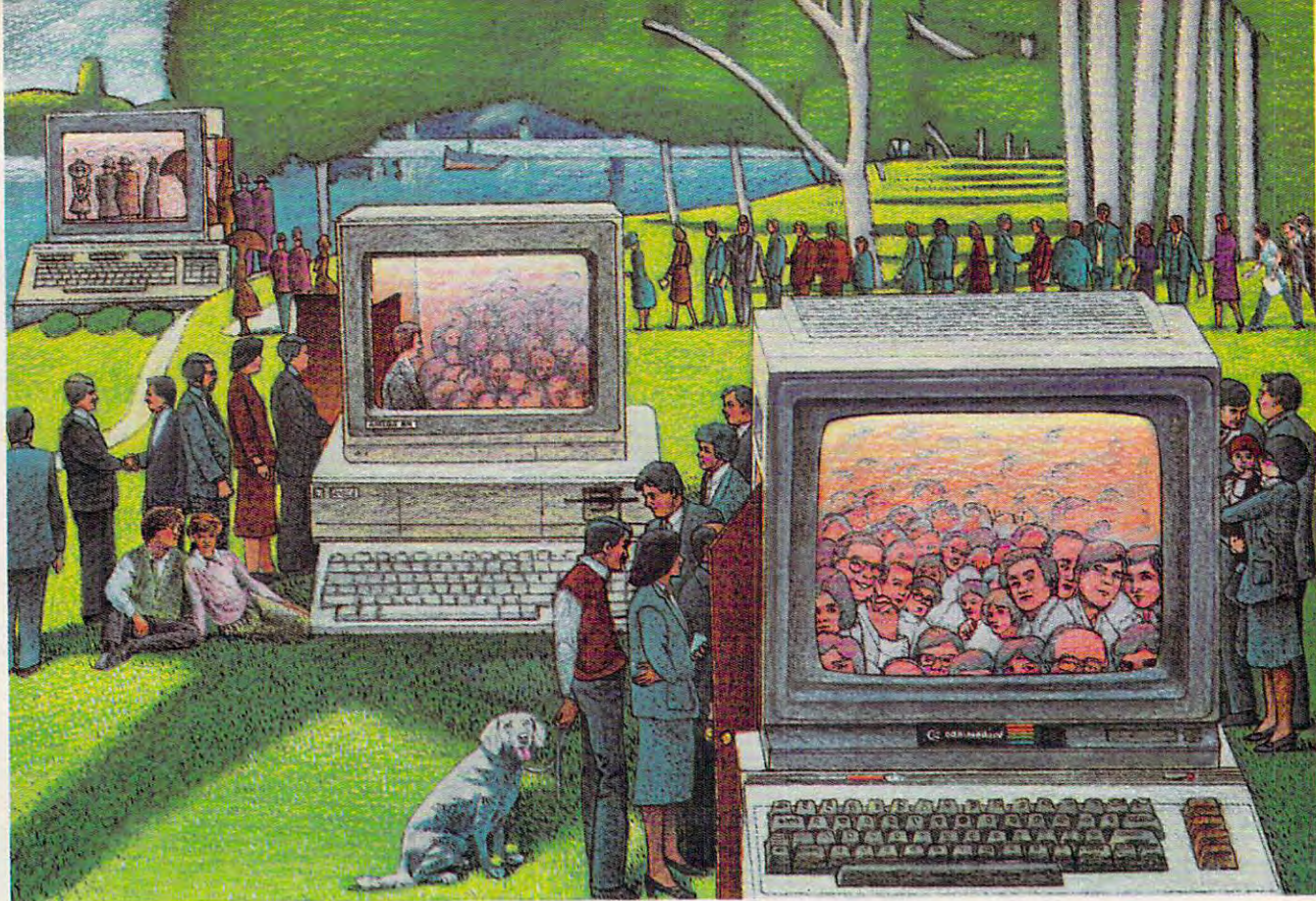
You'll learn electronics, avionics, aircraft maintenance, health care sciences, management or logistics—the Air Force will train you in one of more than 200 technical specialties America needs today.

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).





USE THE BRAINS YOUR COMMODORE WASN'T BORN WITH.

Right at Your Fingertips in CompuServe's Commodore® Forums

Our Commodore Forums involve thousands of Commodore users worldwide. These Forums show you just how easy and fun it is to get the most from your Commodore Computer.

The Commodore **Communications Forum** provides the latest news on communications software and advice on how to effectively use your machine for online computing.

The Commodore **Programming Forum** supports programmers and developers of Commodore 8-bit computers.

The Commodore **Arts and Games Forum** is for all Commodore 8-bit computers. Compose or play music, create or retrieve colorful graphics, and download games software.

The Commodore **Amiga® Forum** is the national resource for all business

and entertainment applications in the Amiga community.

Easy access to free software.

- Download first-rate, non-commercial user-supported software and utility programs.
- Take advantage of CompuServe's inexpensive weeknight and weekend rates (when Forums are most active, and standard online charges are just 10¢ a minute).
- Go online in most major metropolitan areas with a local phone call.
- Receive a **\$25.00 Introductory Usage Credit** when you purchase your CompuServe Subscription Kit.

Information you simply can't find anywhere else.

Use the Forum *Message Board* to exchange mail with fellow members. Join ongoing, real-time discussions in a Forum *Conference*—with Commodore luminaries like Jim Butterfield, Jim Gracely, Bob Landwehr and Steve Punter. Scan Forum *Data Libraries* for free software, documentation and contributions from Commodore enthusiasts.

Enjoy other useful services, too. Like electronic editions of your favorite magazines, newsletters and articles, including Family Computing, OMNI Online and the Electronic Gamer.™

All you need is your Commodore computer and a modem...or almost any other personal computer.

To buy your Subscription Kit, see your nearest computer dealer. Suggested retail price is \$39.95. To receive our free brochure, or to order direct, call 800-848-8199 (in Ohio, call 614-457-0802). If you're already a CompuServe subscriber, type GO CBMNET (the Commodore Users Network) at any ! prompt to see what you've been missing.

CompuServe®

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

800-848-8199
In Ohio, call 614-457-0802

An H&R Block Company

www.commodore.ca

COMPUTE!

JUNE 1987
VOLUME 9
NUMBER 6
ISSUE 85

FEATURES

8	A Beginner's Guide to Programming Languages	Tom R. Halfhill
11	BASIC	Todd Heimarck
12	Machine Language	Richard Mansfield
13	C	Philip I. Nelson
14	Forth	Rhett Anderson
16	Pascal	Tony Roberts
18	Modula-2	Tim Mickiff
25	Laser Chess™	Mike Duppong
62	Super Hi-Res Graphics and Sound on the Apple IIs	William B. Sanders

GUIDE TO ARTICLES AND PROGRAMS

AP/AT/AM/64
AP

REVIEWS

56	Murder Party	Neil Randall
57	Gridiron	Ervin Bobo
58	Destroyer	Scott Thomas
60	Defender of the Crown for Amiga	Neil Randall
60	Championship Golf	Neil Randall

AP/64/PC
AM/ST
AP/64/PC
AM
AM/PC

COLUMNS AND DEPARTMENTS

4	The Editor's Notes	Richard Mansfield
52	Readers' Feedback	The Editors and Readers of COMPUTE!
66	Computers and Society: A Nation of Thieves—Responses to Readers	David D. Thornburg
68	Telecomputing Today: Lapping It Up	Arlan R. Levitan
69	Microscope	Sheldon Leemon
70	The World Inside the Computer: Instant Images on Your Apple Computer	Fred D'Ignazio
71	ST Outlook: Journey from the Center of the ST—Part 1	Philip I. Nelson
72	INSIGHT: Atari—The Protection Racket	Bill Wilkinson
74	The Beginner's Page: Arithmetic in BASIC	C. Regena
75	AmigaView: Desktop Video	Sheldon Leemon
76	IBM Personal Computing: A Bit of BASIC	Donald B. Trivette

•
•
•
•
•
•
ST
AT
•
AM
PC


THE JOURNAL

78	64 RAMdisk	Hubert Cross	64
81	Printing Special Characters	George Vogel	AP
84	Amiga Disk-Based Fonts	Daniel L. Stockton	AM
86	Atari NoDOS	Emmanuel Gendrano	AT
88	Fast Fractal Landscapes for IBM	Paul W. Carlson	PC
91	Atari XL/XE Super Editor	Rhett Anderson	AT
93	Resave	Amy Galtman	64
94	Full-Screen Editor for Applesoft	Alex Wong	AP
96	RAMtest 128	Jim Butterfield	128
98	ML Runner	Paul Lindner	64
99	Car Payments	Brian Flynn	PC
101	College Planner	Brian Flynn	PC
104	GraphiDemo for Amiga	Stefan Lindahl	AM
106	Font Loader for Apple ImageWriter	Ed Thompson	AP

65	CAPUTE! Modifications or Corrections to Previous Articles
108	News & Products
120	COMPUTE!'s Author's Guide
122	COMPUTE!'s Guide to Typing In Programs
125	MLX: Machine Language Entry Program for Commodore 64 and 128
129	MLX: Machine Language Entry Program for Apple
132	Advertisers Index

NOTE: See page 122 before typing in programs.

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

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. Entire contents copyright ©1987 by COMPUTE! Publications, Inc. All rights reserved, ISSN 0194-357X.

In this issue we're proud to present *Laser Chess™*, a fascinating strategy game. Inspired by—but far more futuristic than—traditional chess, this captivating game can be played only on a computer. The winning entry in our \$10,000 programming contest for COMPUTE!'s Atari ST Disk & Magazine, it has now been translated into Amiga, Commodore 64, Apple II, and Atari XL/XE versions.

Elsewhere you'll find a variety of useful utilities and applications. For Commodore 64 programmers, "64 RAMdisk" is highly useful in program development, and "ML Runner" turns machine language programs into files that load and run like BASIC files. "Full-Screen Editor for Applesoft" makes it far easier to program Apple BASIC. "Atari XL/XE Super Editor" significantly improves the screen editor in these machines, and "Atari NoDOS" can restore programs which seemed hopelessly lost. "Fast Fractal Landscapes" for the IBM PC and compatibles offers impressive graphics detail and speed, and "College Planner" and "Car Payments" are compact, easy-to-use programs that help with financial planning. "GraphiDemo for Amiga" is a fast and colorful program which shows off the tremendous processing power and graphics of the Amiga. And "Amiga Disk-Based Fonts" is a helpful utility with a tutorial on how to load custom text fonts from disk and install them from BASIC.

Readers who have been with us for some time are likely to agree that the quality of the articles and programs in this issue does not come as a surprise; rather it's part of a tradition established by Robert Lock, the founder of COMPUTE! Publications.

This past December Robert withdrew from the daily management of the company and accepted the position of Editorial Consultant. He will continue to provide the company with guidance and will assist, in particular, in the development of new products and services.

This change in editorial manage-

ment gives us our first opportunity to publicly acknowledge his contributions to the growth and success of the company he founded and to the personal computing industry as a whole.

In all these years he has never given an interview or in any fashion moved his personality into public view. While some other industry leaders appeared more concerned at times with their personal image than with the health of their companies, Robert has always worked quietly behind the scenes. We can, however, with this editorial, recognize his contribution and thank him in a small way for his efforts.

He was in his early thirties when he had the idea of starting a magazine devoted to consumer computing. And in the past seven years, COMPUTE!—under Robert's guidance—has grown into a highly successful publishing group: four magazines, over 150 books in print, and over 1,000,000 readers a month. COMPUTE! Publications has become a major contributor to the ongoing introduction of computing into the homes, schools, and businesses of America and elsewhere in the world.

He saw early on that computers would have an immense impact. Starting in a storefront in Greensboro, with a handful of employees, he began working on the early issues of COMPUTE!. At that point, he was personally involved in every aspect of his young magazine: He pasted up galleys, called advertisers, contacted authors, and edited copy.

As the years went by, he continued to directly supervise the growing company in both its editorial and business activities. He has an intuitive grasp of business and finance combined with strong editing and writing skills. This combination of talents is as rare as it is powerful. And perhaps even more rare: He is an extremely clear thinker. Those of us who work closely with Robert have always been aware that he is remarkably accurate in his assessments on a wide range of topics.

We at COMPUTE! were not concerned during the notorious industry shakeout of 1984–85. We knew we would be among the survivors. In fact, COMPUTE!'s Gazette was introduced in the face of the shakeout, and became one of the strongest startups in magazine publishing history. We have gone on to publish a number of bestselling books and continue to feature some of the best programs, columns, and articles available on the subject of home and recreational computing.

It all began with a single idea, Robert's vision of how best to engage, entertain, and inform the reader about a powerful emerging technology that he saw would eventually affect every aspect of our lives. But a single idea, however accurate, rarely leads to the creation of a major publishing house. Thousands of others also began to see the importance of consumer computing by the end of the 1970s, and hundreds of publications were introduced. By 1983 there were 150 computer magazines competing for the attention of the reading public. Today there are only a few. That COMPUTE! Publications survived and flourished was largely due to Robert's strong leadership. The staff at COMPUTE!, and the readers who enjoy our efforts, are fortunate that he will continue to play a vital role, contributing to the direction of our publications as we grow.

In the coming years we will, with his help, preserve the traditions and values Robert established here. And we will expand, offering more comprehensive coverage as well as maintaining the quality of programs, tutorials, and features you've come to expect from COMPUTE! Publications.



Richard Mansfield
Editorial Director

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 ISBN 0-87455-068-8
Book/disk combination for the Commodore 64
\$29.95 ISBN 0-87455-069-6

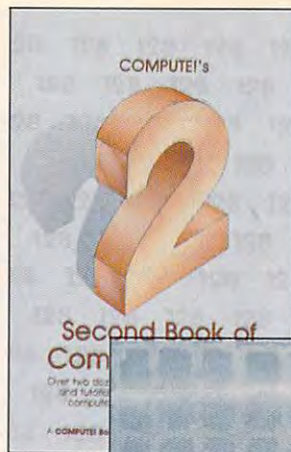
This introductory text to standard Pascal on any computer is an ideal tutorial for anyone who wants to learn this powerful computer language. It includes everything you need, including an introductory Pascal interpreter* for the Commodore 64 and 128 in 64 mode, ready to type in and use. Written in plain 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.



COMPUTE!'s

Second Book of
Com

Over two dozen
and taking
computer

A COMPUTE! B

PASCAL
FOR
BEGINNERS



By M. C. Lutz

A practical guide and tutorial to mastering Pascal on any personal computer. Includes an introductory Pascal interpreter for the Commodore 64 and 128, ready to type in and use.

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. 
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.

 www.commodore.ca

Publisher James A. Casella
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 Editor John Shadle

Assistant Features Editor Kathy Yakal
Programming Supervisor Patrick Parrish
Editorial Programmers Tim Victor, Tim Midkiff, William Chin

Copy Editors Karen Uhendorf, 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
Receptionist Anita Armfield

Associate Editors Jim Butterfield
 Toronto, Canada
 Fred D'ignazio
 Birmingham, AL
 David Thornburg
 Los Altos, CA
 Bill Wilkinson

Contributing Editor Bill Wilkinson

COMPUTE!'s Book Division Editor Stephen Levy
Assistant Editors Gregg Keizer, Tammie Taylor
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

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

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 Atari 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

Weather Wizard For The IBM PC, Commodore 64, Atari, Apple II, Amiga, and Atari ST

Vanishing Directory For The IBM PC

Sprite Designer For The Commodore 64

Medium-Resolution Autorun For The Atari ST

P/M Magic For The Atari

1571 Utility Package For The Commodore 128

IBM Printer Control

Zookeeper For The Amiga

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

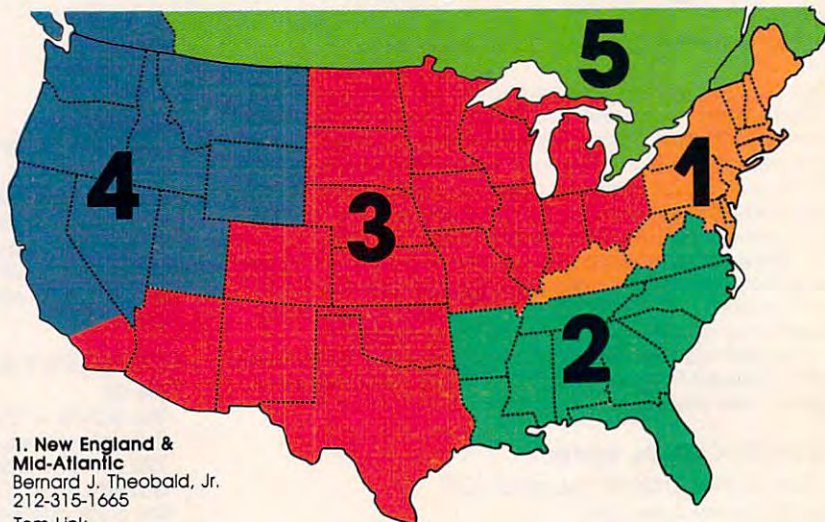
COMPUTE! Subscription Rates (12 Issue Year):

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



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

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

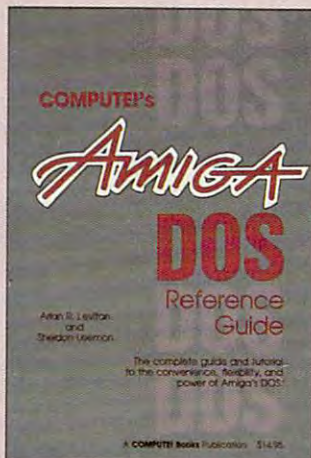
Authors of manuscripts warrant that all materials submitted to COMPUTE! are original materials with full ownership rights resident in said authors. By submitting articles to COMPUTE!, authors acknowledge that such materials, upon acceptance for publication, become the exclusive property of COMPUTE! Publications, Inc. No portion of this magazine may be reproduced in any form without written permission from the publisher. Entire contents copyright © 1987, COMPUTE! Publications, Inc. Rights to programs developed and submitted by authors are explained in our author contract. Unsolicited materials not accepted for publication in COMPUTE! will be returned if author provides a self-addressed, stamped envelope. Programs (on tape or disk) must accompany each submission. Printed listings are optional, but helpful. Articles should be furnished as typed copy (upper- and lowercase, please) with double spacing. Each page of your article should bear the title of the article, date and name of the author. COMPUTE! assumes no liability for errors in articles or advertisements. Opinions expressed by authors are not necessarily those of COMPUTE!.

PET, CBM, VIC-20 and Commodore 64 are trademarks of Commodore Business Machines, Inc. and/or Commodore Electronics Limited. ATARI is a trademark of Atari, Inc. Apple, Apple II, and Apple IIs are trademarks of Apple Computer Company. TI-99/4A is a trademark of Texas Instruments, Inc. IBM PC and PCjr are trademarks of International Business Machines, Inc. Radio Shack Color Computer is a trademark of Tandy, Inc.

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.

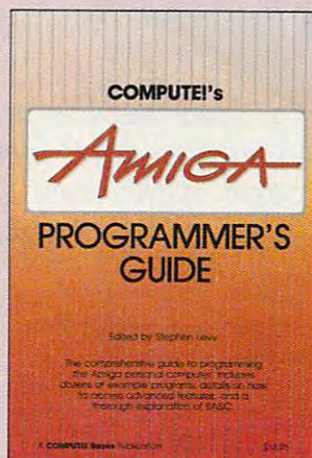
\$16.95 ISBN 0-87455-047-5

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.

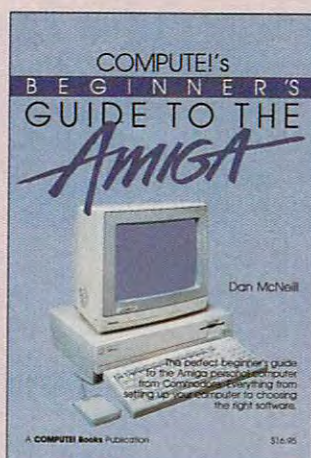
\$17.95 ISBN 0-87455-028-9

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.

www.commodore.ca

A Beginner's Guide To Programming Languages

Tom R. Halfhill, Staff Editor

How hard is programming—really? And which language is best? Here are some straight answers to some of the most commonly asked questions about programming on personal computers.

Recently a teacher whose school was being equipped with personal computers for the first time related how the faculty was introduced to the idea:

"The computer expert for the school district came down and grouped us all together in a room. None of the teachers but me had ever worked with a computer before. The expert proceeded to give us a crash course in computers—but started off by trying to teach us BASIC programming. Pretty soon he was talking about numeric variables, string arrays, and subroutines, and got everybody lost. When the session ended and we left the room, I heard one teacher mutter, 'That's the last time I'm ever going *near* one of those things!'"

These teachers weren't the first people to be discouraged by the complexities of programming. Over the past several years, thousands of students, too, have been exposed to the same approach to computer literacy. Thankfully, most school systems have now recognized the difference between computer *users* and computer *programmers* and have restructured their computer-literacy courses accordingly.

There was a time when a person had to become a mechanic to drive an automobile, become a pilot to fly, and set up a darkroom to take pictures. But as cars, airplanes, and cameras advanced beyond the primitive stage to become mature machines, it became less necessary to wrestle with their inner workings. Likewise, you don't have to be a programmer to operate a computer intelligently and usefully.

Ten years ago, programming was almost unavoidable if you wanted to use a microcomputer. Commercial software was practically nonexistent, and the handful of public domain programs available were usually considered starting points for your own modifications. But

now the personal computer is a mature enough machine so that the average person shouldn't have to struggle with string arrays, subroutines, and recursive procedures. You can probably buy the software you need right off the shelf, type it in from a magazine or book, download it from a bulletin board system or information service, or obtain it from a user group library.

Still, there comes a time when nearly every computer user considers the question: Should I learn to program? Maybe now, as a knowledgeable user, you've realized that computers really aren't as mysterious as they first seemed, so you're wondering if programming might be within your grasp, too. Or maybe you're the type who *likes* to tinker with inner workings—you won't be satisfied until you have total control over the machine. Or maybe you have an unusual application in mind and just can't find the right software to do the job.

If you fit any of these descriptions—or if you're merely curious—you may want to take the plunge into programming.

Primitive Languages

First, a definition: *A program is simply a list of instructions for the computer to carry out.* Some instructions may tell the computer to load a file from disk; others may tell the computer to display something on the screen. It's your job, as a programmer, to write these instructions in the proper sequence. The "proper sequence," of course, depends on what you want the program to do.

So far so good. Now all you have to do is learn what the instructions are and how to fit them together. And this is where things start to get sticky.

Beginning programmers are sometimes dismayed when they discover that the instructions recognized by computers are much more

We've got you covered
on land on sea



C= Apple \$30.00



C= Apple \$30.00



C= \$35.00

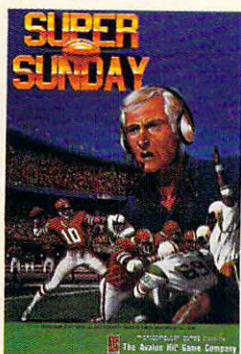
in the air

and on the court



Coming Soon:

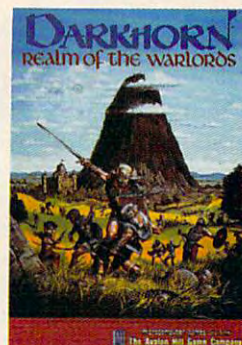
C= Apple IBM



C= Apple IBM \$35.00

on the field

and in your mind!



C= Apple \$30.00

Available at leading game and computer stores
or call TOLL FREE 1-800-638-9292 for ordering information.



microcomputer games DIVISION

The Avalon Hill Game Company

A MONARCH AVALON, INC. COMPANY

4517 Harford Road, Baltimore, Maryland 21214 • 301 254-9200

www.commodore.ca

primitive than they anticipated. COMPUTE! has received letters that read something like this: "Can you tell me which command lets you move a sprite around the screen with a joystick and lets it fire shots?" Or, "Which command sorts a list of names in alphabetic order?"

There are answers to these questions, but they fill enough books to support a large part of the computer publishing industry.

The problem is that individual programming instructions do relatively little. It takes a fair number of them strung together to make the computer do something minimally useful or even recognizable. Imagine, for instance, sending someone to the store for a loaf of bread. Instead of simply saying, "Please go to the store and get a loaf of bread," you have to tell the person, "Walk one block to the corner and turn left. Then walk one more block and enter Joe's Market on the right. Pick up a loaf of bread and pay for it at the counter. Then walk one block back to the corner, turn right, and walk another block home again."

This kind of detail is necessary even when programming in a so-called *high-level language*, like BASIC, Pascal, Modula-2, or Logo. Programming in a lower-level language like C, Forth, or, ultimately, machine language, would be like telling the person how to walk by putting one foot in front of the other, how to open and close a door, how to turn, how to recognize a loaf of bread and pick it up, and how to pay for it.

Computer languages are a little more advanced than the grunts of primitive cavemen—but not much.

Don't let this discourage you, though. The relative simplicity of a computer language makes it much easier to learn than a modern human language. English, French, and German consist of many thousands of words and many dozens of grammatical rules. BASIC, Pascal, and Forth—even the most powerful versions—rarely have more than 100–200 words and a few dozen rules. And, just as your working vocabulary in English is only a fraction of the total language, your working vocabulary in BASIC or Pascal may amount to fewer than 30 commands about 90 percent of

the time.

Someday it may be possible to program a computer in English, French, or German. Researchers at this very moment are working on a *natural language interface*. But until computers get a lot smarter, it's going to be easier for us to learn their languages than for computers to learn ours.

Sorry, No Esperanto

Your first step, then, is to decide which computer language you want to learn. You might be surprised to discover that there are about as many computer languages as there are human languages. If you were born and raised in an English-speaking country, you probably grew up speaking English. However, that same kind of choice isn't made for you when you buy a computer, even if BASIC is built in. Your computer can run several different languages.

Why are there so many computer languages to choose from? After all, it's perfectly logical that hundreds of different human languages developed. There was no communication between different peoples for centuries, and by the time someone came up with the idea of inventing a standard language like Esperanto, it was far too late to get everyone to agree. But computers are a recent invention, so it seems like somebody could have set the standard at the beginning and saved us a lot of trouble.

It's not that simple, though. The reason why there are scores of computer languages is that each one was conceived to solve certain kinds of problems, to run on certain kinds of computers, or to reflect certain programming philosophies. For instance, BASIC—which stands for Beginner's All-purpose Symbolic Instruction Code—was invented by a pair of university professors in the mid-1960s who wanted a language easy enough for neophytes to learn in a few hours. Likewise, Pascal was invented by a Swiss professor who sought a language that was ideal for students. FORTRAN (FORmula TRANslator), one of the oldest languages, was invented for scientists and engineers. COBOL (COMmon Business-Oriented Language), another ancient computer tongue, was invent-

ed for writing business programs. The U.S. Department of Defense even has its own computer language, Ada.

Most languages have also sprouted several dialects. There are dozens of versions of BASIC, for example. Although these dialects are largely similar, there are still too many dissimilarities to allow all but the simplest programs written in one dialect to run under another. Sometimes an extensive rewrite is necessary.

Like choosing a computer, choosing a language is an emotional issue for some people. If you start asking around for advice, expect to hear conflicting arguments about which language is best, and why.

Back To BASICS

All in all, a good starting point for the beginning programmer is BASIC—if only because it's the language you probably already own. Most personal computers come with BASIC on a system disk or built into read only memory (ROM). A BASIC programming manual was probably included in the package, too. If not, a trip to a bookstore or library will turn up dozens of volumes of BASIC manuals.

Besides the fact that it's free, BASIC is a very flexible language that's ideal for experimenting and writing short programs. If you play around with BASIC for a few weeks and decide that programming isn't much fun, at least you haven't spent the \$50 to \$500 that another language may cost.

As an alternative, some languages are available in public domain versions for the nominal cost of a user group library disk or download charges from a bulletin board system or information service. Forth and COMAL are prime examples.

To help you get started, the following pages cover the features of some of the most popular languages available on today's personal computers. This isn't intended to be a complete list, of course, but it will introduce you to some of the pros and cons of several of the most widely-used languages. If you consider the options carefully before making your choice, odds are you won't be disappointed.

BASIC

Todd Heimarck
Assistant Editor

No matter which computer you buy, you'll almost certainly be given a version of the BASIC language with your system. It might be in Read Only Memory (ROM), on a plug-in cartridge, or in a program on the system disk, but whatever form it takes, it's there. BASIC is the *lingua franca*, the universal language, for microcomputers.

Everybody has BASIC. You can write a program, save it to disk, and give a copy to a friend or put it up on a local bulletin board system. Magazines and books publish BASIC program listings which you can study to learn new techniques. Many user groups, schools, and other organizations offer classes in BASIC programming.

If you decide at some point in the future to buy a new computer with more memory or a faster processor, you'll have a head start on programming it if you know BASIC. There are some differences between the manifold dialects of BASIC, but there's a core of commands that are the same on all computers. PRINT, FOR-NEXT, IF-THEN, and many other keywords function virtually identically in all versions of BASIC.

There are other languages available, and they have their good points. BASIC isn't perfect and you'll hear proponents of other languages making comparisons between their favorite language and BASIC. You *don't* often hear someone say, "C is a good language because it has many features that Forth lacks." That's because BASIC is the standard against which others are judged. The most popular language, it has survived for 25 years and probably will still be in use 25 years from now.

Interactive And Flexible

One of BASIC's most attractive features is that it's interactive. You can sit down at a keyboard, type in a few lines, and then type RUN. There's no need to wait while you save the source code, compile and link it, and hope that it works. You just type RUN. The program will either run or it will stop with an error message. BASIC is generally good about pointing out mistakes and telling you which line isn't working. If there's an error, you can list the line and figure out what's wrong.

BASIC encourages experiments. You can stop the program, change part of it, and see the results immediately. A weekend programmer can be creative—adding a line here, changing a variable there—much like a sculptor molding a lump of clay. When you're creating a graphics screen or playing with sound effects, BASIC's flexibility makes it easy for you to test new ideas.

Since BASIC is a high-level language with English keywords, program listings are understandable. You can usually read through a subroutine and follow its logic.

Spaghetti And Celerity

People who dislike BASIC generally offer one of two arguments against it. First, BASIC is an *unstructured* language (imposes few rules on the programmer), and programs often have so many GOTOS that reading the listing is like untying spaghetti. Second, compared to those written in other languages, BASIC programs can run slowly.

Advocates of structured programming suggest that programs should be broken into modules of less than one page each. Variables should be declared at the beginning of each module. Loops should be indented. You should include many comments that explain what's going on. And you should never use an unconditional GOTO.

If you agree that there are advantages to structured programming, you can easily follow all of those rules in BASIC. The modules are subroutines. Variables can be defined at the start of a program.

FOR-NEXT loops can be indented. You can include REMs and avoid the GOTO command. You can write highly structured programs, if that's what you want.

In other words, BASIC doesn't force you to write spaghetti code. It's possible to create a program that looks nice when it's listed. If you see a program that looks messy, blame the author—not the language.

On the other hand, you could ignore the whole idea of structure. If a program works, it works. Some very good programs don't look clean and structured on the inside, but they get the job done.

What about the speed of BASIC? If flat-out speed is what you want, then machine language (ML) is the only choice. All languages are slower than ML.

It's true that some languages are faster than BASIC. But that doesn't matter in some situations. If you're calculating a mortgage payment, for example, BASIC might take three seconds, versus one-third second in a faster language. If you had those extra seconds, what would you do with them?

In other cases, the problem isn't BASIC, but the algorithm used. For example, a bubble sort that alphabetizes a list of 500 names might run very slowly. You could rewrite the program in another language and it would run the bubble sort faster. But if you changed to a Shell sort or a quick sort, you'd notice a definite improvement in speed, even in BASIC. There are many techniques for speeding up BASIC programs.

To make comparisons fair, BASIC programs should be *compiled* (run through a special process which translates them into something resembling machine language). A compiled BASIC program often runs at the same speed as a similar program in another language. With a compiler, you can have the best of both worlds.

Machine Language

Richard Mansfield
Editorial Director

En oino aletheia, if you don't know Greek, is, as they say, Greek to you. Likewise, `CMP #5FD:RTS` looks pretty strange if you don't know machine language. But there's nothing inherently complicated about either language—once you know the vocabulary and the rules.

Too many BASIC programmers stay away from machine language (ML—sometimes also called *assembly language*) because they take one look at it and it appears impossibly alien. In fact, it is simply another language with about 50 words to learn and some new techniques. The fundamentals of programming in any language remain the same: data lists, loops, branching, subroutines. So, once you've learned to communicate with your computer in one language, you've got a considerable head start when learning a second language.

Machine language *does* require somewhat more exacting programming (in the sense that cooking dinner from scratch is more exacting than opening cans). But there are many tools now available to the ML programmer: single steppers, error-detecting assemblers, and a wealth of books which help the novice move quickly through a bafflement phase toward eventual mastery.

Talking Directly To The Machine

But why, if you already know another language, would you want to learn ML? One clue is that most commercial or professional programming (especially on computers such as the Commodore 64) is ML. This is because ML is the *machine's language*. Any other language requires translation before the computer can understand what to do. For example, when you use the BASIC command `PRINT`, the computer looks up the print subroutine.

(This subroutine is, like the subroutines in virtually any computer language, written in ML.) The lookup is one level of indirection, but there's also the problem that the `PRINT` statement can have so many different forms (are we printing to screen? is there a `TAB`? and so on). All this translating takes time and, with interpreted languages like BASIC, this time is spent *while the program is running*.

The primary benefit of ML is that there is no lag time between the computer's reading the instruction and its ability to act upon the instruction. ML, the computer's own language, speaks directly to the microprocessor chip. So-called *higher-level languages* all require, to one degree or another, compromises and indirect, time-consuming, superfluous tasks. The reason for this is obvious: The computer can speak *only* one language—ML. Anything else you offer it will have to be reduced to ML before the computer can respond.

Two higher-level languages renowned for speed and power—C and Forth—illustrate this point. Both languages are interesting, enjoyable to program, and a significant improvement in efficiency over BASIC. However, after programs are written in C or Forth, they are often further modified by replacing key sections with pure ML. There are even utilities which watch an executing C program and then report where the computer is spending most of its time—where in the program are the most delays. If the report reveals that the computer is spending 40 percent of its time in lines 256–288, you rewrite that segment, replacing C with ML.

A Subtle Intimacy

The most important issue, then, is speed, and nothing can work faster than ML. For years BASIC programmers, too, have included ML sections within the BASIC program to make game animation smoother, spreadsheets calculate faster, or word processors able to keep up with quick typists. We can see this endless quest for additional speed in hardware as well: clock speeds have gone from 1 MHz to 16 MHz. In the world of computers, there's

no such thing as having too much memory or being too quick.

There's an additional advantage to ML which is significant, but somewhat subtle: intimacy. With ML you are working with a *low-level language* (the only low-level language), and that means you're right down in the engine room and you've got all the tools you need to do anything you want. In other words, you're down on the computer's level, and you can see and access all its features. For example, it's often not possible from higher-level languages to test whether the `BREAK` or `CONTROL` keys are being pressed on the keyboard; to design disk directory lists which suit your needs; to redefine keys; to speed up input; or to quickly switch entire screens of information. Most of these tasks are easy to accomplish via ML; all are possible.

A machine language programmer works with instructions which, by themselves, accomplish less than many of the instructions of higher-level languages (`PRINT` is made up of hundreds of ML instructions). This has two primary consequences: You use more instructions per ML program, and, therefore, have more bugs to fix. It takes somewhat more time to program ML. (Your reward is that your finished program will run perhaps 100 times faster than its high-level-language equivalent.) A second consequence is that ML is freer than other languages, more uniquely the work of one individual. For this reason, higher-level languages are often chosen for team programming—the separately programmed parts are more likely to combine harmoniously into a finished product when a more rule-oriented, less individualistic language is used.

But, if you work alone, enjoy a challenge, and want to write the very best programs your computer can run, you might want to give ML a try. It's not really more difficult than other languages—just different.



Philip I. Nelson
Assistant Editor

You might call C a "middle-level" language, since it is easier to write and understand than machine language, but it lacks some convenience features found in high-level languages. C originated in the early 1970s at Bell Labs, where it was first used to transport the UNIX operating system from a DEC mini-computer to other computers.

Today, C is used to write everything from arcade games to operating systems for new computers. C is a favorite of commercial software houses for two main reasons: C programs compare favorably with machine language programs in performance, and C is not tied to any particular operating system, making it suitable for programs that must be translated for several different computers. And as reasonably priced C compilers have made their way into the home computer market, C has become popular with hobbyist programmers, too.

Keyword Economy

C has fewer than 30 keywords in all, and about half of those are used to declare variables and other data objects. The most important remaining keywords are

if, else, case, switch, default, for, do, while, break, continue, return, goto

If you're used to BASIC—with versions having anywhere from 50 to 200 keywords—you might wonder how any useful language can get by with so few words. Part of the answer lies in the fact that C never uses two words where one will do. For example, where BASIC uses FOR and NEXT to create a FOR loop, or WHILE and WEND to create a WHILE loop, C dispenses with NEXT in the first case and

WEND in the second. The occurrence of *for, while, or do* is sufficient to identify a loop in C, and this economy of expression typifies the language in general.

Another reason why C requires few keywords is that it relies heavily on context and uses many words and operators for more than one purpose. For instance, the type identifier *int* can have quite different meanings depending on where it occurs in a program. In a variable declaration, *int* means *I plan to use an integer variable with the following name*. When it occurs before a function, *int* means *The following function returns an integer value*. And when it appears in a cast, *int* means *Convert the following noninteger value into an integer*.

C has a luxurious supply of operators. In a simple example, BASIC uses the statement $a = a + 1$ to add 1 to the value of the variable *a*. C permits the same syntax, but it can do the job more economically, in two different ways:

```
a++;  
a += 1;
```

Several variable types are available in C. You must declare every C variable—state its name and its type—before using it; this requirement, among other things, makes C programs easy to read, since variable declarations can occur in only a few places. The preferred style is to use *external* variables—those known to every function in the program—only where necessary, thus minimizing the risk that a secondary function will inadvertently garble a variable used elsewhere in the program. Most C variables are *local* to one degree or another, meaning that they are known only to a single function or related functions.

Another powerful data object is the *pointer*—a variable that points to some other object. A common use of pointers is to access individual elements of an array. However, pointers have many other uses, including direct manipulation of the computer's memory.

An array in C is a collection of data objects of the same type under one name. For example, a string is an array of objects of the type *char* (character) which ends with a zero.

More complex data objects include the *structure*, which is a collection of objects of dissimilar types under one name. Each *member* of a structure can be accessed by a unique name, although the entire structure is treated as a logical unit. Not only can a structure accommodate both arrays and simple variables within the same skin, but it also can hold variables of different types—characters, integers, floating point numbers, and so on.

C data types may be combined quite freely, allowing you to create arrays of pointers, pointers which point at pointers, arrays of structures, pointers to structures, self-referential structures, and so on. This fact, combined with C's general economy of expression, permits you to create concise, elegant programs.

Easily Transportable

The C language does not provide input/output (I/O) functions of any kind (it has no keywords such as READ or OPEN, for instance). This is in keeping with the notion that C should not be shackled to any single computer or operating system. However, every C compiler includes a so-called standard library of common I/O functions which do tasks such as printing text, reading and writing files, and so forth. A program that uses nothing but standard I/O functions should be transportable, with little or no change, to almost any computer for which there is a C compiler. However, this ideal is rarely achieved except for very plain, generic programs, since most programmers will want to take advantage of special features unique to the host computer. To simplify the process of translating programs, C programmers often segregate machine-specific code in distinct, easily identifiable modules.

Forth

Rhett Anderson
Assistant Editor

Every programming language has its friends and enemies, but perhaps no language stirs up as much controversy as Forth.

Forth's notoriety is a result of a significant distinction between it and other popular computer languages. While most languages provide a set of standard commands and functions, Forth encourages you to extend the language itself to fit the task. It's been said that Forth is a "toolbox," and that you have to build every program from spare parts. Once you've written several programs in Forth, you've built your own set of tools to make writing future programs easier.

Forth was designed by one person—Charles Moore—to control astronomical instruments. Its speed and size, however, have made it a versatile language for microcomputers: It is fast and compact.

Forth is a low-level language. This means that when you program in Forth, you are very close to programming in the computer's native language. Because of the dense structure of Forth, large Forth programs may be even smaller than equivalent machine language programs. Forth allows easy access to a machine's hardware. As a matter of fact, it's as easy to read or write to a hardware port as it is to read or write to a variable.

Forth is often one of the first languages to appear on a computer because it is so easy to write a Forth interpreter. Sometimes there are good Forth implementations even before there are good BASIC implementations.

A Difference In Notation

People who are used to languages like BASIC and Pascal are usually shocked when they see their first Forth program. This is partly due to the fact that Forth uses *postfix* notation instead of *infix* notation.

Infix notation is the method of calculating that we learn in school. It is the notation used by Texas Instruments calculators and the BASIC language. Here is an example:

$(1 + 2) * (5 + 4)$

This expression evaluates to 27. Parentheses are used to specify which calculations are to be performed first. In infix notation, you always have to keep the order of operations in mind—for instance, multiplication and division are done before addition and subtraction (unless parentheses indicate otherwise). As you might imagine, computers are slowed down by taking all these rules into account.

In postfix notation (used in Hewlett-Packard calculators and the Forth language), the same expression would look like this:

$1 2 + 5 4 + *$

Postfix notation has the advantage of never requiring parentheses. The expression is evaluated from left to right—always.

Another characteristic of postfix notation is that numbers are put onto a *stack*. A stack is simply a place to hold numbers. It is a LIFO (Last In / First Out) structure. This means that if you put a 4 and then a 5 into the stack, you'll get the 5 back before the 4. Math is always done this way in Forth.

Postfix notation is much faster on a computer than infix notation. Most microprocessors have special commands to handle stacks. Some even have built-in hardware stacks. This helps make Forth very fast indeed.

Besides its heavy use of stacks, the critical difference between Forth and other languages can be found within the structure of Forth itself. New words (which are equivalent to procedures and subroutines of other languages) are defined as a series of pointers to previously defined, lower-level words. The path of execution of a Forth program

follows these pointers from high-level words down through lower-level words until at last a *primitive* word is reached. (A primitive word is one which is defined in machine language instead of Forth.) Following this path is like untying a ball of kite string, but the computer manages the task with breakneck speed.

Some of the original arcade games were programmed in Forth, games that would be nearly impossible to program in BASIC.

Forth is fast as an interpreted language, but there are still a few Forth compilers available. However, compilers don't speed up Forth as much as they do BASIC.

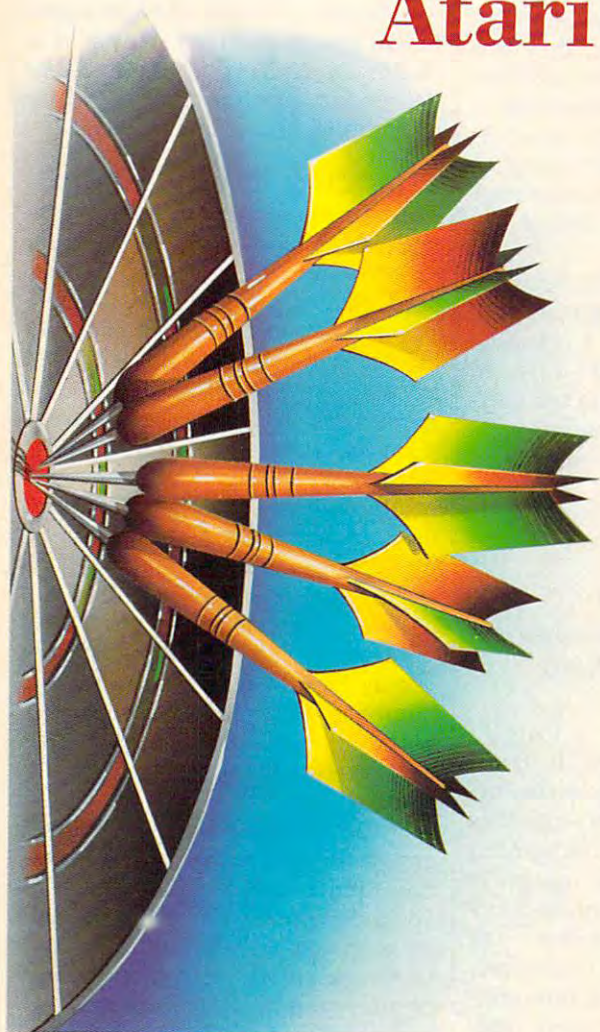
Small, But Powerful

Forth's small size was especially important in early computers. Some versions of Forth have been written to use less than 4K of memory. Now that computers have more RAM, new Forth packages emphasize new features. Recent versions of Forth for the ST, Amiga, and IBM are very powerful languages which allow full file access, floating point numbers, and multi-tasking.

There are disadvantages to Forth, too. It can be difficult to read, write, and debug. Early implementations did not have floating point numbers, and Forth advocates are still debating whether or not the advantages of floating point numbers outweigh the disadvantages (floating point numbers are slower and use more space). Another problem is file storage. Earlier implementations of Forth could not use the standard file formats of the computers they ran on. But recent implementations have solved this problem, too.

Forth remains a popular language because it offers great flexibility. When you program in Forth, you play by your own rules—it's possible to customize and personalize it more than any other high-level language. Although some say that it's dangerous to program at this level, others demand the freedom that the Forth language offers.

Mindscape Aims To Please With New Hits For the Atari XE™/XL™ and ST™.



BALANCE OF POWER



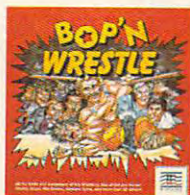
by Chris Crawford. You are the President of the United States. Tension always escalates in this global geopolitical simulation of the cold war's cruel reality. Test your savvy with what the *New York Times* called one of the most sophisticated strategic simulations in America. Available on the **Atari ST** only.

INFILTRATOR



Fasten your shoulder harness and seat-belt. *Infiltrator* by Chris Gray puts you at the controls of a powerful gunship ready for action. You are Johnny "Jimbo Baby" McGibbits with miles of enemy airspace to clear. You'll then convert to covert ground action, behind enemy lines, to destroy the mad leader and his hostile troops. Available on the **Atari XE/XL** only.

BOP 'N WRESTLE



Pow! Sock! Slammo! *Bop 'N Wrestle* is as close as you can get to professional wrestling without drawing blood. Put yourself into the ring with 10 of the biggest, baddest bruisers ever to perfect the turn-buckle fly. Available on the **Atari XE/XL** only.

TRAILBLAZER



Push yourself to the limit in this hyperspatial spherical grand prix. Leave your opponent in a cloud of dust as you leap over black holes, purple walls, blue bouncers, and the terrible cyan nasty zone. Not for the faint at heart. But surely an action arcade adventure in which you can have a ball. Available on the **Atari XE/XL and the Atari ST.**

HIGH ROLLER



Strap yourself into a Harrier jet and enter the dangerous world of vertical takeoffs, barrel roll attacks, and heat-seeking sidewinders. Learn to fly the most sophisticated combat aircraft in this flight simulation software. Your mission is to destroy the enemy headquarters 500 miles away. Good luck! Available on the **Atari ST** only.

Mindscape
Software that challenges the mind.



Infiltrator is available on Apple II family, IBM & compatibles, C64 & C128 and Atari XE/XL. *Bop 'N Wrestle* is available on Apple II family, IBM & compatibles, C64 & C128 and Atari XE/XL (128K recommended). *Trailblazer* is available on C64 & C128, Atari XE/XL and Atari ST. *Balance of Power* is available on IBM & compatibles, Atari ST, Macintosh, and Amiga. *High Roller* is available on C64 & C128 and Atari ST.

Visit your retailer or call 1-800-221-9884 (in Illinois 1-800-942-7315) 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.

If you're an attorney read this: Apple, IBM, PC Jr., Commodore, Atari and Amiga are registered trademarks of Apple Computer, Inc., International Business Machines, Commodore Electronics Ltd., Atari, Inc. and Commodore Amiga, Inc. respectively. Mindscape is a trademark of Mindscape, Inc.

www.commodore.ca

Pascal

Tony Roberts
Production Director

For those whose natural tendencies veer toward disorder, computer programming can degenerate from a pleasant occupation into a chaotic nightmare in as little as a few short GOSUBs.

Pascal, with its structure and its rules, can help keep programmers out of this trouble, as well as make their jobs more rewarding and their programs more useful.

Pascal, developed by Niklaus Wirth in the 1970s as a teaching language, is named in honor of the seventeenth-century philosopher and mathematician Blaise Pascal. Wirth reasoned that if a language's source code were clear and understandable, and if its compiler could identify errors, the language would be an efficient teaching tool. Programmers who learned Pascal would have a solid understanding of programming and could easily pick up other languages as needed.

As it turned out, Pascal grew beyond the classroom and has become a development tool for many programmers who need a language that's easier to work with than machine language, but faster than BASIC.

Anyone familiar with BASIC will see many similarities between it and Pascal. Pascal programs manipulate variables and arrays and use program-control structures such as FOR-NEXT, IF-THEN-ELSE, and WHILE-WEND loops.

Pascal differs from BASIC, however, in that it imposes specific rules about how a program is to be constructed. A Pascal program consists of the following items, which must appear in the order indicated:

- Program identifier
- Constant, type, and variable declarations
- Definition of functions and procedures
- The main program

Help Or Hindrance?

Pascal's detractors complain that the rigid rules and set structure stifle creativity; those who have grown fond of the language lean heavily on its structure for support. Using the framework required by Pascal, proponents claim, they can spend more of their time deciding how the program will *work* rather than how it will be *written*. In addition, the rules help the compiler detect and flag errors, which in turn helps the programmer debug the program.

For example, Pascal requires that all variables be *declared* before being used. Declaring a variable simply involves stating the variable name and the type of data the variable can hold. The following variable declarations might be used in an adventure game:

```
VAR
  score      : integer;
  level     : integer;
  strength  : real;
```

Now, assume that the program contains the following lines:

```
score := strength;
lever := level + 1;
```

When the program is compiled, both lines will generate errors. In the first case, the variables are of different type. You cannot assign the value held in *strength* (a real variable) to the variable *score* (an integer variable) because the types are incompatible. In the second line, notice that the programmer has accidentally typed *lever* instead of *level*. Because the variable *lever* has not been declared, the compiler will report that *lever* is an unknown variable. Upon seeing this, the programmer immediately recognizes the typographical error and corrects it. This type of inconspicuous error can be elusive in BASIC.

Pascal programs are usually written as a series of subroutines that are called, as needed, by the program's main body. Essentially the goal of a Pascal programmer is to break down every task into a series of subtasks, writing a separate subroutine to handle each of the subtasks.

These subroutines, called *procedures* and *functions* in Pascal, are designed to be relatively independent. That is, they can be tested and

debugged separately from the rest of the program. Making a change in a procedure or function should only affect the small task being performed by that procedure or function.

Subroutines are just like mini-Pascal programs: They consist of a name; constant, type, and variable definitions; function and procedure definitions; and a main body.

Extending The Language

Procedures and functions are the building blocks with which you both construct a program and extend the language itself. Once a subroutine works in *Program A*, you can use it in *Program B* as well. Let's say you wrote the procedure to convert a string to uppercase and print it on the screen. You might call it `Print_Up_String`. This procedure could then be compiled into and used in any future program you might write. It's the equivalent of adding a new keyword to BASIC.

A mathematician might develop a set of subprograms that efficiently handle certain calculations. Someone who works extensively with data files would develop procedures for opening, closing, and reading files.

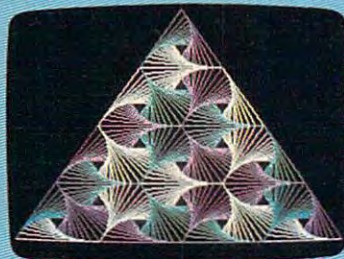
This ability to reuse completed and debugged subprograms saves time and builds the confidence of any programmer. Once you've built up a library of subroutines, you'll be more comfortable attacking larger, more sophisticated projects.

Scope Of Variables

Another way Pascal differs from BASIC is in *scope*, which is the range in which a variable is effective. In BASIC, variables are *global*; that is, they can be referenced from anywhere in the program. Pascal can use global variables, but the language also makes heavy use of variables with lesser scope, and this contributes to the ease with which procedures and functions can be used as program building blocks.

When a variable is declared at the top of a Pascal program, it is a global variable, and it can be referenced or manipulated anywhere in the program. When a variable is

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.

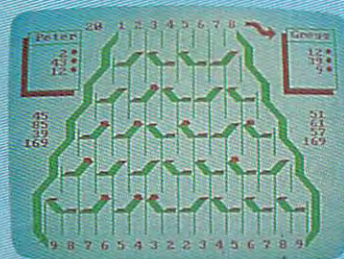
Subscribe now and you can depend on a steady supply of high quality, fun-filled programs like Hickory Dickory Dock, Switchbox, TurboDisk, Home Financial Calculator, Turbo Tape, SpeedScript, SpeedCalc, and hundreds of other educational, home finance, and game programs the entire family can use all year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

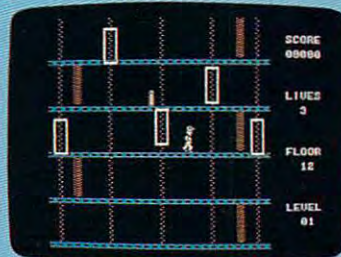
Whether you're a novice or an experienced user, COMPUTE! is the magazine for you. So subscribe today. Return the enclosed card or call 1-800-247-5470.

Month	Expenses	Income	Profit
January	1812.22	592.22	422.79
February	422.22	1842.22	628.00
March	5821.00	4387.22	32.78
April	1456.22	2976.22	1477.81
May	512.22	2474.22	1311.79
June	1223.11	2384.22	381.11
July	241.45	2888.00	1656.55
August	617.00	1922.00	1285.00
September	1221.00	1250.00	747.00
October	412.44	8888.00	7587.56
November	677.37	2888.00	1222.13
December	2121.00	2888.00	378.00
TOTAL 5 MONTHS	14448.15	41448.15	26723.56
AVERAGE:	1427.14	1836.13	2463.89



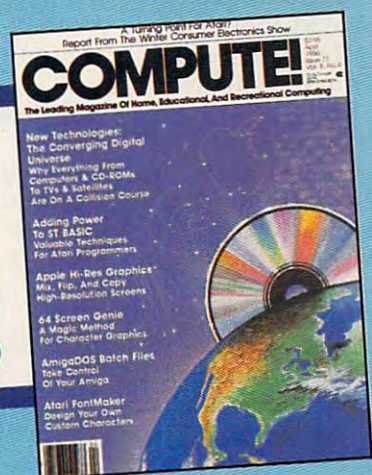
SpeedScript
Writing a machine language program that works on five different computers is no small task. The first hurdle is finding a "safe" place to put the code. Though the cassette buffer is an obvious choice, it's located in different places on various machines, and putting HL there creates problems for tape users. Instead, the Proofreader uses 256 bytes of BASIC programming space. *

Before it installs the routine in memory, the Proofreader checks to see which computer you're using. Then it stores the HL at the bottom of BASIC memory and protects itself by moving the computer's start-of-BASIC pointer to a spot 256 bytes higher in memory. Once that's done, the Proofreader activates the HL routine and erases itself with NEH. Note that because the Proofreader overwrites its first few BASIC lines, it's critical not to delete anything from the first portion



Do it now.

ACT NOW AND SAVE!



COMPUTE! Publications, Inc. 

One of the ABC Publishing Companies

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

www.commodore.ca

declared within a subprogram, its scope is limited to that subprogram. When the subprogram is called, Pascal creates the variable and tracks it, but when the subprogram is exited, the variable is deleted and the space it occupied in memory is made available for other uses.

The same variable—the traditional loop counter *i*, for example—can be used in several subprograms without conflict. As long as the scope of a Pascal variable is limited to one subprogram, you need not worry about using unique variable names.

When To Use Pascal

Pascal can be used for any programming project, but it is most useful for programs that are regularly used. Because they're written in a compiled language, Pascal programs run many times faster than those written in an interpreted language like BASIC. However, it usually takes longer to write a Pascal program than to write a similar BASIC program.

For short utility programs that will be run once—or twice—you're probably better off sticking to BASIC, but if you're writing a program to be used daily in your business, Pascal is a good language to use. The increased speed will be

appreciated by those who use the program, and you, the programmer, will appreciate the ease with which you can make minor changes as needs dictate.

Learning the elements of Pascal is relatively easy, but some of the language's more advanced features, such as the ability to create new data types, to manipulate complex data structures, and to manage large amounts of memory through the use of pointers, require a bit more study. Once you've mastered these topics, however, you'll find Pascal able to handle most programming problems.

Modula-2

Tim Midkiff
Editorial Programmer

Modula-2, introduced in 1980 by Niklaus Wirth, the creator of Pascal, was designed to provide a clear and natural way to solve common programming problems. Wirth also wanted the language to be suitable for large-scale software development and system-level programming (teams of people working on a single program).

Modula-2 provides facilities for looping, branching, and procedures that closely resemble their Pascal counterparts. The languages are very similar, so Pascal programmers can easily adapt to Modula-2.

Compiler Requirements

Modula-2 compilers require that all variables be declared. The standard data types include arrays, enumeration, subrange, sets, records, and pointers. *Type checking* is performed on the usage of variables.

Type checking insures that assignments and expressions do not contain incompatible data types. As an example, assuming that *i* had been declared an *integer* and *x* had been declared a *real* variable (a variable which can include a decimal point and fractions), the compiler's type checking would flag this line as illegal:

```
x := i;
```

Type transfer functions are provided to allow operations that may be illegal under normal type checking. For example, using the same variables as in the above example and a type transfer function, this line would now be considered legal by the compiler:

```
x := REAL(i);
```

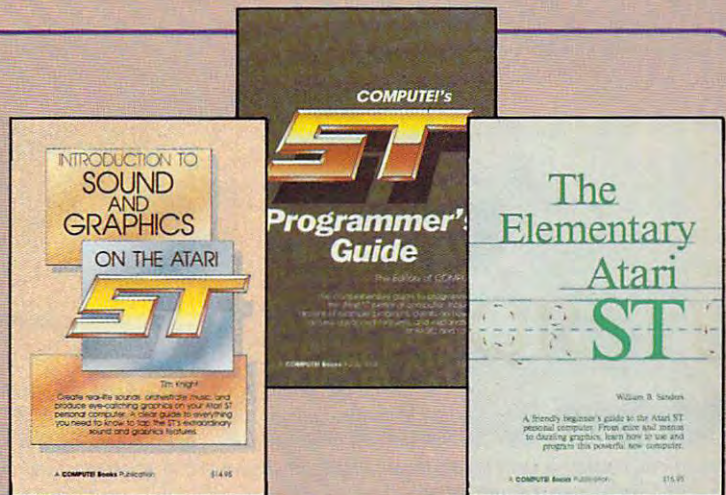
Modules and coroutines provide support for group software development. Modules are separately compiled units which allow programmers to build libraries of code that can be imported by other programs. Modules declare all procedures and data which can be imported by other programs. This facilitates *data hiding*, which prevents inadvertent changes to variables. The coroutine support also allows programs to transfer control to other programs, allowing the software to simulate multiprocessing.

The low-level support provides many functions necessary to access machine-specific features. Absolute addressing allows access to specific memory locations such as ports and system variables. There are procedures which return the address of variables or the amount of memory used to store the variable. Also, there is a procedure which allows the insertion of machine language routines by giving the opcode and operands for an instruction. This is useful to speed the execution of time-critical sections of a program. Bit manipulations are provided as part of the set data type.

Modula-2 is a compact language with only about 40 reserved words. Much of the machine-specific programming is provided in the libraries. There are standard libraries for I/O, strings, files, coroutines, and low-level machine routines. Most implementations of the language for personal computers provide system calls as part of the libraries. This is an important consideration when you're comparing implementations, as it greatly facilitates programming. ©

COMPUTE! Books' ATARI ST Collection

COMPUTE! Books offers you a superior line of titles for the new Atari ST. Packed full of useful utilities, exciting games, in-depth tutorials, and valuable applications, these clearly written books bring you fully tested information and entertainment for the whole family.



Look for these COMPUTE! books at your local book or computer store.



COMPUTE!'s First Book of the Atari ST

Edited

A valuable collection of ready-to-type-in-and-use applications, games, and utilities. Graphics utilities like "ST Doodler," games like "Switchbox" and "Tug-a-War," and educational programs like "Hickory Dickory Dock" turn your Atari ST into everything from a business graphics machine to a powerful teaching tool. Tutorials show you how to add power to ST BASIC and how to add excitement to your own creations with sound effects. A disk is available for \$15.95 which includes all the programs in the book, 203BDSK.

\$16.95 ISBN 0-87455-020-3

The Elementary Atari ST

William B. Sanders, 272 pages

A friendly, easy-to-use guide to the Atari ST, this book takes you through connecting your computer, loading programs, creating graphics and music, and writing your own programs.

\$18.95 ISBN 0-87455-024-6

Elementary ST BASIC

C. Regena, 208 pages

A tutorial and reference guide to the ST's impressive graphics, animation, and sound with complete descriptions of ST BASIC's commands, syntax, and organization. A disk is also available for \$15.95 which includes programs from the book, 343BDSK.

\$14.95 ISBN 0-87455-034-3

COMPUTE!'s Kids and the Atari ST

Edward H. Carlson, 238 pages

Easy-to-understand instructor notes, lessons, assignments, and lively illustrations help both kids and adults painlessly learn to program on the Atari ST. The latest in the bestselling series by this author.

\$14.95 ISBN 0-87455-038-6

COMPUTE!'s ST Programmer's Guide

Editors of COMPUTE!, 356 pages

A comprehensive reference guide to the Atari ST, this book explores in detail Logo and BASIC, the advanced features of the ST such as GEM and TOS, and every aspect of programming from concepts to actual program writing.

\$17.95 ISBN 0-87455-023-8

Introduction to Sound and Graphics on the Atari ST

Tim Knight, 197 pages

Thorough descriptions of the Atari ST's color graphics and sound abilities, plus all the information needed to create a complete sound and graphics system.

\$16.95 ISBN 0-87455-035-1

You can order directly from COMPUTE! by calling 800-346-6767 (in NY call 212-887-8525) or by sending your payment to COMPUTE! Books, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

Please include \$2.00 per book shipping and handling for U.S. and surface mail or \$5.00 for airmail. North Carolina residents add 5 percent sales tax. New York residents add 8.25 percent sales tax.

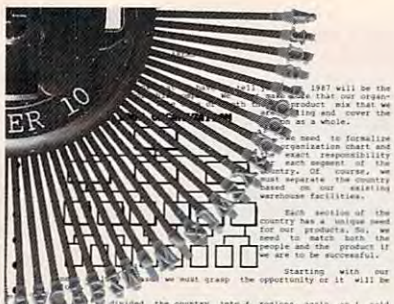
Please allow 4-6 weeks for delivery from receipt of order.

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

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! Gazette, COMPUTE!'s Gazette Disk, COMPUTE! Books, and COMPUTE!'s Apple Applications



1987 will be the year that our organization takes the big and cover the...
 We need to formalize organization chart and...
 Each section of the country has a unique need...
 Starting with our...
 We've divided the country into 4 regions, each as I said...
 But, they'll have to make frequent trips to be sure that the actual direct customer contact is being maintained correctly...
 I would like to receive any suggestions at Friday's meeting that would make the plan go more quickly...
 This is a major move for the company. I value everyone's input. And I expect everyone...



DAK INDUSTRIES INCORPORATED

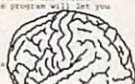
Dear Customer,
 Now you can have text and graphics together on one page with your own logo and even a border of the type to the side or at the bottom. Now you won't have any more problems when you want to print out letters. And what's best of all is that there won't be any steps to worry about in the program either.

Once you're more or less sure how easy it is to add graphics to your letters or reports, and you can add drawings and what to your business letters or proposals.

This program is so easy to use that you will use it from now on. And what's really great is that if you already use a word processing program, you can plug in files that you've written in your other program and use them in this one in minutes.

So no matter where you live in the United States, you'll find that this program will let you really use your mind.
 So, if you do use your brain, you can really get into the swing of using this great new program.
 There is simply no limit to the impact that drawings can provide for your work. It doesn't matter if it's a chart, a graph or picture of a house, you can do it all with this exciting new program.
 So, don't let the tech talk of "disturbed stand" fool you. You don't need any special hardware to make a really first class impression. After all, they say that a picture is worth 1,000 words, just think of how many ideas are represented on this single page of copy.

For formal proposals, for marketing reports or for impressive letters, you can do it all with this powerful new program.



Dear George,
 Well, your sales for the last 7 months have been exceptional. I've light seen the latest graphs, and I must say I'm impressed. Who you said that you could increase sales and keep royalties even. I must admit that I was skeptical.

But, I think this graph says it all. It's also very impressed with manufacturing. We've seen a real benefit in the increased sales in a reduction in fixed costs.

So overall, not only have our sales increased, but as a percentage of sales, costs have gone down.

May we be the only month that didn't show an increase in sales. But it showed a decrease in costs. So you can bet that our bottom line increased.

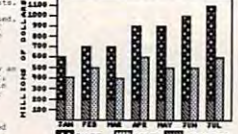
I'm very impressed with the results and I think that the money we spent for automation and to implement your marketing effort was well spent. At year end, we'll really be able to see how long the payoff will take. But, from these early results and the fact that I know you'll maintain your efforts for the last 7 months of the year, I think you'll be able to get some more of the things you want next year.

The second graph on this page is perhaps even more important than the first. As you know we were receiving far too many customer complaints.

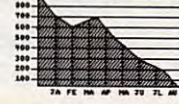
I am very impressed with the reduction in calls that this has had on representing the extra people in quality control and in customer service are really paying off. Considering the cost of complaints, this drastic reduction clearly shows a return on investment to the bottom line.

George, this has been a very good year so far. Let's push to make the last 7 months look as good as the first 7. I know at the board meeting that you'll be a major topic of conversation if this trend continues.

SALES AND COST REPORT



CUSTOMER COMPLAINTS



\$89⁹⁰ Desktop Publishing Breakthrough

Imagine using a word processing and drawing program that lets you integrate charts and pictures that you 'paint' or 'clip' into your text. Well, if you use an IBM PC or Clone, now you can have graphically dramatic documents, from business or personal letters, to proposals, to organization charts, even with a daisy wheel printer.

By Drew Kaplan

It's easy. It's impressive. And, now your thoughts can be powerfully illustrated in both words and graphics.

After all, for illustrating abstract data and thoughts, nothing beats a dramatic chart or drawing. So, let your ideas leap off the page by using integrated text and graphics. Your thoughts are sure to make an impressive impact.

Whether you write letters, bank proposals, term papers, company manuals or news letters, you can forget complicated and expensive laser printing. And, you can forget complicated expensive desktop publishing programs.

Now for just \$89⁹⁰, you can use your daisy wheel, dot matrix or ink jet printer to print normal text. Plus, you can integrate simply fabulous graphs and drawings into your creations.

INCREDIBLY EASY

Savtek, a brain trust group, has developed an easy to use yet incredibly sophisticated integrated word processing and graphics program.

Just create your letters, proposals, or reports as you would with any other word processor. In fact, if you already have a document created in virtually any other word processor, you can 'grab' it into Savtek's instantly.

You'll produce visually powerful technical papers and manuals with drawings and charts, and dramatic marketing reports with graphs. You'll produce sales proposals with panache.

And since there's no complicated training needed (if you can run a word processor, you can run Savtek), you'll make great impressions, fast.

Anyway, once you've created the written part of your report, using Savtek's sophisticated automatic word processing features, you're ready to add pictures, charts and graphs.

Just select from the over 100 supplied changeable pictures or draw your own, using the automated ICON based drawing program.

Later, you'll learn much more about

the sophisticated drawing program that lets you draw, paint, fill, expand, reduce, copy, and move your pictures.

And, you'll form squares, circles and triangles automatically. Anyone can draw with it because it's totally automated and uses arrow keys and doesn't require a mouse. But, read on.

Once you've selected a picture, the computer will produce an automatically sized box representing it. Just position the box wherever you want the picture to be in the text.

Like magic, the actual picture will appear and the text will automatically reformat itself around it.

And, speaking of reformatting, this program will automatically make page-breaks and recalculate each page as you write or edit. If you make an addition to page 1 of a 10 page report, the effect will ripple through all 10 pages.

So, whatever length you've chosen for each page (including headers, footers and automatic page numbering), will automatically be preserved.

You'll particularly like the cut and paste features of this word processing program which allow you to copy, move or delete sections of your text.

Of course, you'll have automatic Word-wrap, Hidden Hyphenation, Justified Smooth Right or Ragged Right text. Plus, you'll have Find, Replace and Search.

And look how you can format your document. There are 5 page templates called rulers which allow you to automatically set up your page.

You can select any right and/or left margins, your tabs, one, two or three line spacing, and the number of blank lines at the top and bottom of your page.

Each of the 5 rulers comes with different default settings. But, you can adjust and save them or change them and even use several at one time on a page.

HOW DO THE PRINTERS WORK?

I use a daisy wheel printer because I like my letters to look personal. I've always had to switch to a dot matrix printer for graphs and illustrations.

... Publishing Continued

It simply removes the last thing you did. So, no matter what you do wrong, you're a button away from removing it.

If you don't want a solid line, just spray an area. It's like using a spray can.

Let's say you want to connect two points with a straight line. Use the Angle Line. It produces a computer generated straight line between any two points.

What if you want a circle? Just touch the return key. Then use the diagonal arrow key to enlarge or reduce the circle. If you use the up/down or right/left arrows, you'll get an ellipse.

In the same way you can create squares, rectangles or triangles. And you'll be amazed how many things, from houses to technical drawings, are made up of squares, rectangles, circles and triangles.

But, that's not all. You can choose any of 32 background patterns to fill in enclosed areas or broad lines. And if 32 isn't enough, you can design your own.

There's so much more. You can juggle a picture. Imagine, turning it over or sideways with the touch of a button.

You can copy or move a picture or even part of a picture right on the screen. So, draw it once and copy it or move it.

But, here's my favorite. You can enlarge or reduce any picture or part of a picture right on the screen. So you can change its size equally, or you can stretch it out or make it tall and thin. Wow!

There are 12 included font/sizes. So you can have large or small type in your choice of styles within a picture or integrated with your text.

And, each of the 12 font/sizes can be shown on the screen and printed normally, in bold, in italic, in outline, or in shadow. Plus, you can write normally across the page, up the page, down the page or upside down.

Finally, you can zoom into any small section of the screen and edit your pictures, pixel by pixel. With this kind of power, you don't need to be an artist, just have the ability to push a button.

You can operate this Paint program independently. Or, you can access any picture from within word processing.

So, for banners and pictures, you can

print directly from the Paint Program. Or, for everything previously described, simply access your pictures, captions, graphs or charts through the desktop publishing section.

This program is incredibly powerful, yet you'll be comfortable using it within just a few hours.

Every picture in this ad was created with this program. And, you haven't even seen the tip of the iceberg of its capabilities. For example, if you have a picture on the screen, you can bring a second picture up and join them together.

WHO CAN USE THE SYSTEM

All you need is an IBM PC, AT, XT or 100% compatible with standard IBM CGA or EGA graphics capability. It must have at least 256K, and either two floppy disk drives or one floppy and a hard disk.

Below is a list of some of the dot matrix, ink jet and daisy wheel printers that have been tested with this program. If your printer is compatible with any of these printers, it should work too.

Special Note: Most daisy wheel printers are Diablo 620/630 compatible, so they will work with this program.

Special Note: With a color printer you can print 3 colors plus black text.

C. Itoh 8510, Epson Fx-80, Fx-85, Fx-185, JX-80 (color), LC-800, LQ1500, LX80, MX80 with Graftrax Plus or Graftrax, RX-80, Hewlett Packard 2225C Think Jet or QuietJet, LaserJet, or LaserJet Plus, IBM 80CPS Graphics Printer, IBM Printer, IBM 3852 Jetprinter (color), Juki 6100, Mannesmann Tally Spirit 80, NEC 3500, 3510, 3520, 3530, 3550, 5500 series, 8023A, NEC Pinwriter P5XL, P6, P7, (single or color), OKIDATA Microline 92, ML92, w/IBM Plug & Play, Microline 193, 20 (color), Panasonic KX-P1091, KX-P1091i, Quadram Quadjet (color), Radio Shack DMP-200, Silver Reed EXP 400, 600, 800 and all EXP series, Star Micronics SG-10, Texas Instruments 855, 865, Xerox (Diablo) 620, 630.

FINAL FACTS

There's a pop down calculator which lets you deposit your results right into your text. A clock/timer picks up the time from your computer, and there's a 7,300 year calendar. They are all available as pop-down windows. Savtek's program is backed by a standard limited software warranty /license. It comes with a superb, easy to use reference manual.

DESKTOP PUBLISHING FOR YOU RISK FREE

Make your ideas explode in front of your readers. When you send out a letter or proposal, let it be really dramatic. They will be your ideas, but you'll be presenting them like never before.

If you're not 100% satisfied, simply return it in unused condition within 30 days for a courteous refund.

To order Savtek's ETG Integrated Word Processing and Graphics Desktop Publishing System for your IBM PC or Clone, call toll free or send your check for the breakthrough price of just \$89⁹⁰ (\$4 P&H) Order No. 4801. CA add tax.

Look at the 12 sample pages I created. You'll see graphs, pictures and charts mixed into my text. I even designed a logo for my newsletter. Just think about the impact you'll make when you present your ideas with a combination of text and graphics. And oh, it's so incredibly easy to use. IBM is a registered trademark of International Business Machines.

DAK INDUSTRIES INC.
Call Toll Free For Credit Card Orders Only
24 Hours A Day 7 Days A Week
CP65 **1-800-325-0800**
For Toll Free Information, Call 6AM-5PM Monday-Friday PST
Technical Information... 1-800-272-3200
Any Other Inquiries... 1-800-423-2866
8200 Remmet Ave., Canoga Park, CA 91304

SMOKING
Bar chart showing smoking trends. Includes text: "I've been getting a lot of letters from people who are asking me how to quit smoking..."

IT'S YOUR ORGANIZATION
Organizational chart showing a hierarchy of departments like Sales, Marketing, etc. Includes text: "I've been getting a lot of letters from people who are asking me how to organize their business..."

DESIGN LAYOUTS
Three sample page layouts for newsletters or reports, showing text and graphics integration.

TOTAL PICTURE CONTROL
Backgrounds section showing various map patterns of the United States. Includes text: "I've been getting a lot of letters from people who are asking me how to make their maps look better..."

CHAPTER 2
A large stylized number '2' with decorative elements. Includes text: "I've been getting a lot of letters from people who are asking me how to make their chapter headings look better..."

THE NEWSLETTER HELP NOW
A page with various icons and text for newsletter help. Includes text: "I've been getting a lot of letters from people who are asking me how to make their newsletters look better..."

TOTAL CREATIVE CONTROL
A page with a heart-shaped graphic and text: "HAPPY VALENTINE'S DAY". Includes text: "I've been getting a lot of letters from people who are asking me how to make their Valentine's cards look better..."

BEAUTIFUL PAGES
A page with decorative borders and text: "THANK YOU!!", "HAPPY BIRTHDAY". Includes text: "I've been getting a lot of letters from people who are asking me how to make their thank you notes look better..."

IDEA STARTERS
A page with various small illustrations and icons for idea starters. Includes text: "I've been getting a lot of letters from people who are asking me how to make their idea starters look better..."



Bold Presentations

Unleash the dramatic presentation power of your IBM PC or Clone, with Quadram's 7-Color Ink Jet Printer. You'll produce attention grabbing professional graphs, charts, and drawings with the complete ADDED software DAK has put to-

- Quadram's \$495 Color Ink Jet Printer
- Softkey's \$149 Keychart
- DAK's \$49⁹⁰ DAKGraf
- DRI's New \$299 Gem Draw Plus
- Savtek's \$89⁹⁰ Desktop Publishing
- Logitech's \$119 LogiMouse Plus

gether. It connects instantly. And, you get **EVERYTHING**, a \$1201⁹⁰ value, for just \$499. WOW!

By Drew Kaplan

Let your ideas explode in front of your readers. Demonstrate what you want to say with dramatic charts, graphs, pictures and even technical illustrations.

If you're like me, columns of numbers are boring. Now you can effortlessly produce sophisticated, multicolor graphs and charts that will rivet your reader's attention to your presentation.

Turn your thoughts, calculations or sales results into a profusion of color that simply can't be ignored.

This Quadram 7-Color, virtually silent, ink jet printer, and all the software I have added, will let you make your most complex concepts easy to understand.

And, it will present your easy to understand numerical data and/or visual concepts in an incredibly impressive manner.

WHAT YOU NEED

You'll need an IBM PC, XT, AT or Clone with 256K of RAM, except for Gem Draw Plus which requires at least 384K. You need two floppy disk drives or a floppy and a hard disk.

And, all these programs require only standard IBM CGA Graphics (EGA is great of course). Plus, you don't even need a color monitor to use them.

WHAT IS INK JET PRINTING?

In a few minutes you'll learn how easy it is to instantly create multicolor Pie, Line and Bar Charts with DAKGraf. With Keychart you can quickly create sophisticated Hi/Low Stock market Charts, Scatter Charts, Exploded Pie Charts and a myriad of presentation charts from easy to follow menus.

If you want to Paint, you can create all

types of dramatic pictures and text automatically with Savtek's Desktop Publishing Software.

And finally, Digital Research's new Gem Draw Plus lets you create the most sophisticated drawings ever. It is object oriented so you can even lay shapes on top of one another and then move them, expand them, reduce them or copy them.

Plus, I've included Logitech's C7 Serial Mouse with Plus Software. So, not only can you use this mouse for creating drawings, just wait till you use it for Lotus or any other mouse supporting programs.

Logitech says that the Plus Software with this mouse can increase the productivity of even the most experienced 1-2-3 user by up to 30%.

In virtually every computer magazine I see, Logitech sells this mouse direct for \$119. Wait till you see all it can do for your productivity.

But, let's look at the printer. It simply plugs into the parallel output of your computer. And, don't worry about replacing your printer.

I've got an inexpensive option that lets you switch from your regular printer to this silent 7-color ink jet printer with the push of a button.

Anyway, speaking of 7 colors, let me tell you what they are and how easy they are to use. First, there are no messy ink trays or bottles. The ink is supplied in 2 sealed, plug-in plastic packs (Included).

One pack supplies 4,000,000 characters of black. The second pack supplies 3,000,000 characters of yellow, blue (cyan), and red (magenta).

These are the industry standard basic

colors. The other colors are obtained by the printer using two colors at once.

So, you'll print in black, yellow, red, blue, green (yellow and blue), orange (yellow and red) and purple (blue and red). You'll simply have to see the colors to see how really vibrant they are.

And speaking of vibrant, if they aren't sharp, they aren't vibrant. And, this printer can lay down 640 single dots per line at a speed of 2600 dots per second.

You'll create incredibly sharp detail, even in multicolors. Just look at the main picture above to get a minuscule idea of what this printer is capable of producing.

You can produce your graphs, pictures, drawings, or text (of course it has a full ASCII character set, plus enlarged type) on just about any material you choose, from letterhead, to plain paper, to roll paper, to overhead transparencies.

Quadram's printer has all the expected features, including Line Feed, Form Feed, paper error/end and two self-test modes. But basically, it's control free.

Just plug it in, put in the paper and print! It's virtually maintenance free because the ink jets are self cleaning.

As you'll see from the following software, there's almost no limit to the presentations you can create with this printer.

And please note: I went to 5 separate companies to gather this software and mouse. This isn't cheap bundled software.

Just price Gem Draw Plus at \$299 and the Logitech Mouse at \$119 at any computer store, and you'll see just what an incredible value this system is.

We had to guarantee not to sell the

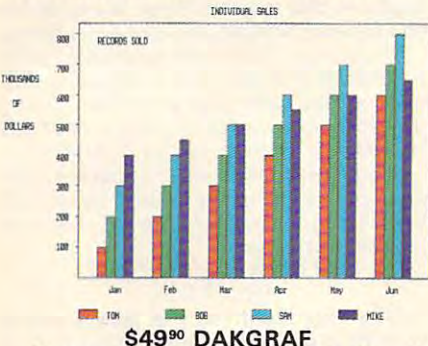
Next Page Please
www.commodore.ca

...Presentations Continued software separately and we had it shipped to us without the normal fancy retail decorative binder/boxes.

Note: I understand that Keychart was once sold with Quadram's printer, but we bought it direct from Softkey.

Anyway, other than the boxes, you get the same complete programs. There are drivers for dozens of other printers, every word in every manual is included, and every part of each program is included.

Now, let's take a look at all the things you can do. If you're already familiar with any of the software, you'll know that the following descriptions are simply the tip of the iceberg of the presentation graphics resources that you'll command.



\$49⁹⁰ DAKGRAF

Now you can create powerful, vibrantly colored graphs that really get your meaning across. You'll be producing graphs in seconds. I find it easier to spot trends with a graph than with a spreadsheet.

Let's make a simple one element graph. Just enter up to 12 numerical values (called observations). Then touch G. Then select a Pie, Line or Bar graph.

Instantly you'll see a full preview of your graph. But, there's lots more.

If you touch ESC, you can re-select another type of graph and it will instantly appear, using your existing data. So, you can choose the type of graph you want.

Anyway, if you've selected Line or Bar charts, the tick marks and values (called scaling) that you see on the left side of most graphs, can be added automatically.

Across the bottom of your chart, you're going to want to assign names to the 12 observation points we've chosen.

The names may be people, things or months. Just touch ALT M, and the Months will appear automatically. Finally, you can add or move floating captions.

Pie charts are also very easy. When the pie chart prints out, it can either show the actual number beside each slice or the percentage of the whole.

So far, we've made a simple one element graph. But for Line and Bar charts, you can have 4 elements. So, at each observation, you'll have up to 4 different colored bars or lines. You can compare profits, costs, sales and salaries.

Below the graph, a box with the appropriate color is printed. Just type a title next to it to identify each line or bar.

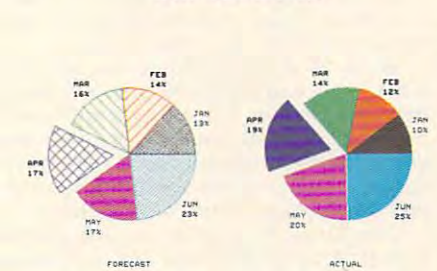
Of course, you can save your graph. You can recall it and change it. You can even change it from a bar, to a line, to a pie, in any order you like. And, you'll be running this program faster than it took to read this description.

When you print out your color graphs, you can choose from 4 different sizes.

\$149 KEYCHART

Here's the ultimate charting presentation program. It can run circles around the DAKGRAF. Not only can you create regular charts, you can explode pie slices and print in any of 7 different hatch designs in addition to the 7 colors.

WATERLOO TECHNICAL SERVICES
FORECAST SALES VS. ACTUAL
KEYCHART PRESENTATION GRAPHICS



But simple Pie Line and Bar charts are too easy for this program. It's very easy to use and completely menu driven.

In addition to being able to preview your graphs on screen, you can even edit your chart's size, location or you name it. Look at all the charts you can produce.

You can make Bar Charts, Clustered Bar Charts, Stacked Bar Charts, Horizontal Bar Charts, Line Charts, Combination Bar and Line Charts, Pie Charts, Multiple Slice Exploded Pie Charts, Scatter Plots, Combination Line and Symbol Charts, X-Y Charts (Business & Scientific Formats), High-Low-Close-Open and Volume Stock Charts, Area Fill Charts, Regression Analysis Charts, Log-Log and Semi-Log charts, Text Plots, Combination Chart and Text Plots.

All these charts may look complicated, but they are really easy to use. There's a standard template menu and all you do is fill in the blanks.

And look at this. You can import electronic spreadsheet information from programs such as Lotus®, MultiPlan®, SuperCalc®, VisiCalc® and more. So, you can fill in the blanks or import the data.

Dear Jim,

I've just received the info. on the new breakdown of the sales territories. I think that setting this change will result in the way we distribute our goods.

Jim, if we do break down the territories into three regions I think we will be much more successful. Just how we have seen the best use of our existing resources.

I think that each person will be much more able to take care of their own area. And, this should mean faster sales and good news to me. I'll be looking forward to reviewing your thoughts.

SALES AND COST REPORT

You can see that our sales have grown, so I think that it's time for us to make the best use of our people. Look at the left and you'll see the improvements that have taken place.

Let's move forward with the new plan and we'll see just how good it can be. I know you'll like the results on each side.

Last, I think it's time for us to consider the offer that was made by the western company that we approved. With the new breakdown of territory I think we can handle the situation.

Between the new building that we have planned and the latest improvements in our service, I'm sure we can do it all.

Jim, the combination of both and whether you really let me know the decision (the being pleased) is not a matter of straight, but you know this is the best way to go.

Well, you can consider all the possible ways for us to proceed this.

\$89⁹⁰ DESKTOP PUBLISHING

Savtek's Desktop Publishing program is incredibly easy to use. It's an integrated text and graphics program.

So, you can combine your text and pictures on the same page, but not on the same line.

Let me tell you about the paint program. It is absolutely a dream to use. It comes with over 100 predrawn pictures, but creating your own is easy.

And, if you're not an artist, don't worry,

everything is automated. It uses the arrow keys to draw. It's very fast and you'll be amazed at how easy it is to use.

You can draw lines in any of 12 different width/styles. And, you can instantly erase them with the Eraser function.

If you don't want a line, use Haze. It's like using a spray can. It lightly mists the area you move over. The more you go back over an area, the more dense it becomes. It's great for shadows and fill.

Want a straight line? Just choose two points and 'Line' will do the rest.

What if you want a circle? Just touch the return key. Then use the diagonal arrow key to enlarge or reduce the circle. If you use the up/down or right/left arrows, you'll get an ellipse.

In the same way, you can create squares, rectangles or triangles. And you'll be amazed how many things, from houses to technical drawings, are made up of squares, rectangles, circles and triangles.

Now lets have some fun. This program is incredibly powerful. Let's say you've created a square. You can pick it up and move it anywhere on the screen that you wish by using 'Move'.

Let's say you really like your square. Use 'Copy', and you can make as many copies as you like. And you can place each copy wherever you like. This is really great for organization charts or anything where you need repetitive shapes.

But, we're not even close to being finished. What if you don't like the size of what you've drawn?

'Vary Size' lets you enlarge or reduce any object on the screen. You can even stretch it out or make it tall and skinny. I stretched out a map of the United States.

And, there's still more. You can juggle a drawing on the screen. You can turn it over, around or sideways.

Finally, you can Zoom in on a section and 'operate' on it pixel by pixel for infinite control of each dot.

OK, now for some thoughts on color. There are 32 different fill patterns. You can see them in the main picture next to my map of the United States. Each of these patterns can be altered.

You can create your own patterns. So, you can make your logo, happy faces or you name it. Whatever you choose, you can automatically fill in any closed area.

This program will allow you to paint in 3 colors at a time. You can draw in black, blue and red, or yellow, blue and red.

You can also form patterns that are combinations of the colors for even more variation. There's also a provision for full size and half size picture printing.

My favorite part. OK, I'm not the world's greatest artist and I make mistakes (lots of them). There's an UNDO command. Whenever I mess up, I just touch UNDO, and my last action is instantly undone.

Finally, there are 12 font/sizes so that you can have headlines, titles or text within any of your drawings.

And, each of the 12 font/sizes can be shown on the screen and printed normally, in bold, in italic, in outline, or in shadow. Plus, you can write normally across the page, up the page, down the page or even upside down.

This is an incredible program. And, don't forget, you can integrate the pictures into its sophisticated word pro-

. . . Presentations Continued
cessing program instantly.

\$299 GEM DRAW PLUS

Now, let's get really serious. While Savtek's paint program is superb, I think Gem Draw Plus is the most sophisticated drawing program in the industry.



Its power is virtually unbelievable. Making squares, rectangles, circles, arcs, and polygons, are mere child's play.

Expanding, shrinking or copying elements of your picture is accomplished with the click of a mouse (more later).

You can draw with up to 16 colors (this printer will print 8 including white). Each color is numbered for use on non-color monitors.

And, you can use 35 fill patterns. And each can be used with any color.

Gem Draw Plus is 'object oriented' rather than pixel or 'screen' oriented. It understands what you want to create, so it keeps the components (It calls them 'elements') separated in its memory.

So, for example, if you overlay a circle with a square, they mix on the screen, but not in the memory.

You can put one behind the other and then switch them. This program never forgets the objects you're working with.

If you design a house, you may make a toilet. Then you may make 10 copies. Later you may want to make them smaller.

Just gather them all together into a 'group', reduce them, and then 'ungroup' them and put them back wherever you want. It's great for architects, engineers or designers.

It's particularly strong for finance, manufacturing and higher education. Of course, you can draw sophisticated pictures just for fun. But, whatever you draw will be technically perfect.

And perfect is an understatement. Look at all the ways you can align the elements of your picture with just the click of the mouse. You can: Put In Front, Put In Back, Make Group, Break Group, Align Left, Align Center, Align Right, Align Top, Align Middle, Align Bottom, Page Center, and Even Spacing. Wow, all this is from just one pull down menu.

The Make Group and Break Group is incredible. If you've created a number of parts to your picture, a single command lets you combine them so you can do something to as many of them as you've chosen. Or, you can separate elements and act on each individually.

The list of drawing aids goes on and on, including auto-grid, and I certainly can't cover them all here. But look.

Let's say that last week you created a drawing. It could be an electrical schematic, an organization chart, or a forest full of trees. Now you want to create a new drawing. But, you want to 'pick up'

some of the parts of your old drawing.

After all, the best part of computers is that you never have to do the same work twice. Well, Gem Draw Plus allows you to bring up two separate screens at one time. So, for example, you can have your new picture on the left and the old picture on the right side of the screen.

OK, now it gets exciting. You can 'drag' an element from your old picture across the screen into your new picture.

Wow, so if you do repetitive types of work, you can instantly pick up parts from old pictures to save yourself time.

Of course, you can alter the element you've moved just as if you'd just drawn it. And, you can move something you've just drawn back to the old picture.

Gem Draw Plus gives you incredible power. And its graphics are especially compatible with Ventura® Publisher.

There are multiple line sizes with choices such as arrows, straight or rounded endings. There are different size fonts, of course. There's a library of artwork and there's a Shadow Command that gives any object a 3-dimensional look.

Gem Draw Plus comes complete with GEM Desktop which is a utility program that provides the 'Gem Environment'.



\$119 LOGIMOUSE PLUS SOFTWARE

It's keyboard freedom when you plug Logitech's C7 serial mouse into your RS232 serial port. I've included it because it makes Gem Draw Plus so easy to use. But, you'll use it all the time.

Just plug it into your serial port and get ready for super productivity. This is the advanced version with special software that really speeds up your work.

The Point-and-Click software make the popular Lotus 1-2-3 work like a mouse-based application. It fully integrates the mouse, making it easier to create and edit spreadsheets. With Logimouse and its Program, you can scroll to different sections of your spreadsheet and move quickly from cell to cell.

It has its own time-saving pop up menus, which you can customize to meet your needs. With its Point Text Editor, you can open many overlapping windows on the same or different files.

You'll find that a mouse added to your keyboard will make your work infinitely easier in lots of programs.

What is a mouse? Well, it's very simply a small device you move on your desk. As you move it, it causes the cursor on the screen to move.

It replaces keyboard commands and is incredibly fast. It lets you be really productive. When the cursor is where you want it, simply click (touch) a mouse button and your computer will react.

You will find more and more programs supporting mice because they are incredibly easy to use.

I think you'll have a hard time matching the quality of the Logimouse. And when you add the Plus Software, I can't match the productivity at any price.

WHY SO CHEAP

This system will come to you in just a few boxes. But, it took me over two months to assemble the software. I had to work with 5 separate companies, plus Quadram, to make it complete.

The problem was really very simple. Nobody at Quadram knew what software was available for this printer.

So they were sitting with 4200 printers. I bought all 4200 for a song and put together this package.

The only reason that the price is so 'cheap' is because I got a ridiculous price from Quadram. They are a large company with lots of other products, and 4200 printers wasn't worth their effort.

From Boston to Toronto to Silicon Valley, I've covered this continent to put together this system. **Now, it's easy for you to use because you get everything.**

Of course, if you just wanted to print text, virtually every word processing program works great. But you'd be wasting the incredible presentation power of this remarkable 7-color ink jet printer.

It's backed, as is each separate software package and the mouse, by each of the 6 individual manufacturers' standard limited warranties.

BOLD PRESENTATIONS RISK FREE

Don't say it with words. Demonstrate your ideas with vivid colors. Show graphs to make your point. Design presentations to knock the socks off of prospects. Or, simply draw beautiful pictures.

If you aren't 100% awed, simply return it in its original boxes within 30 days for a courteous refund.

To order the Complete Presentation package including the \$495 Quadram 7-Color Silent Ink Jet Printer, DAK's \$49⁹⁰ DAKGraf, Savtek's \$89⁹⁰ Desktop Publishing Program, SoftKey's \$149 Keychart, Digital Research's newest \$299 Gem Draw Plus, and Logitech's \$119 Logimouse with Plus Software for a total \$1201⁸⁰ retail value, risk free with your credit card, call toll free or send your check for just \$499 (\$14 P&H) Order No. 4811. CA res add tax.

Special hookup bonus. Now you can switch between your current printer and this printer instantly. Just unplug the cable from your parallel printer and plug it into this box. Two identical cables are supplied that connect our switching box to both of your printers. It's just \$49⁹⁰ (\$3 P&H). Order No. 4813.

A box of 4 125' rolls of 8½" wide paper is \$29⁹⁰ (\$3 P&H) Order No. 4486.

Extra 4,000,000 character Black Ink Packs are just \$12⁹⁰ (\$1 P&H) Order No. 4484. Extra 3,000,000 character Yellow/Red/Blue Ink Packs are just \$14⁹⁰ (\$1 P&H). Order No. 4485.

You never get a second chance to make a good first impression. With this system, every impression will be dramatically bold and 100% professional.

IBM PC & XT & AT, Gem Draw Plus, Keychart, Savtek, Quadram, Logitech & Logimouse & Point and Click, and Lotus 1-2-3 are registered trademarks of International Business Machines, Digital Research, Softkey, Savtek, Quadram, Logitech and Lotus Development Corp. respectively.



**DAK INDUSTRIES
INC.**

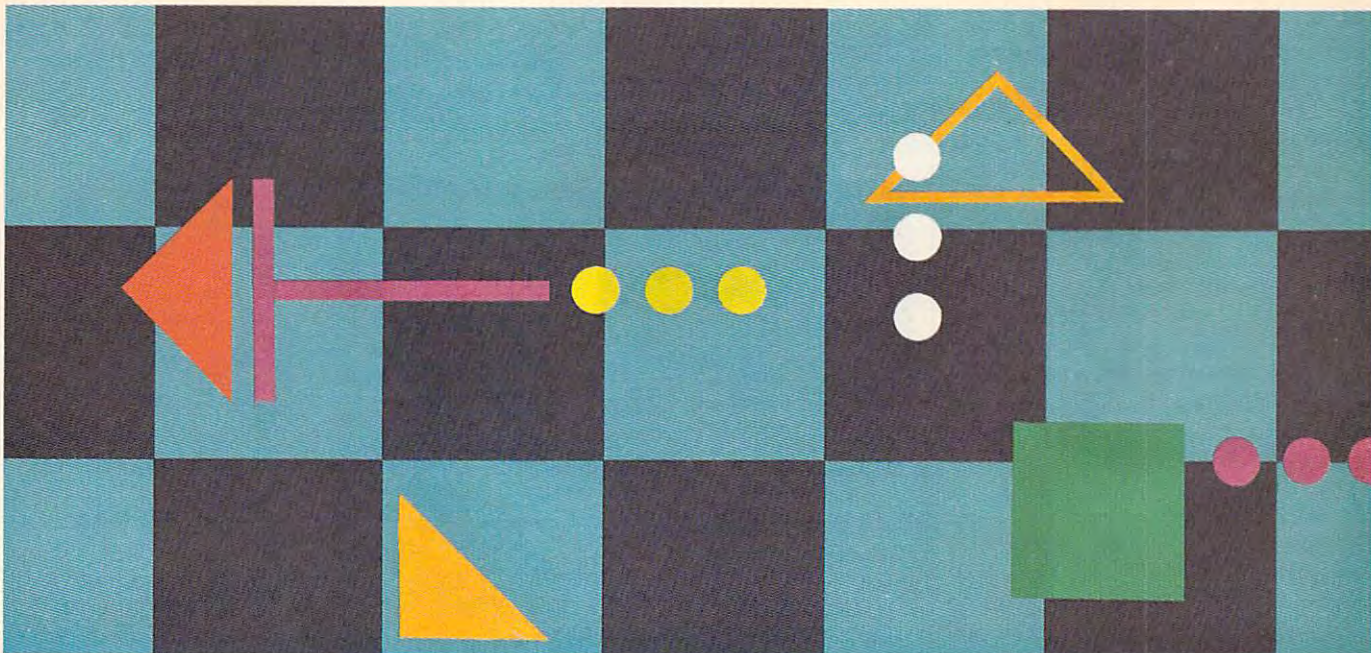
Call Toll Free For Credit Card Orders Only
24 Hours A Day 7 Days A Week

CP70 **1-800-325-0800**

For Toll Free Information, Call 6AM-5PM Monday-Friday PST
Technical Information. . . . 1-800-272-3200
Any Other Inquiries. . . . 1-800-423-2866

2200 Remmet Ave., Canoga Park, CA 91304

www.commodore.com



Laser Chess™

Mike Duppong

Here's a game that's so good that we just had to share it with COMPUTE! readers. Laser Chess™ won First Prize in our \$10,000 programming contest for COMPUTE!'s Atari ST Disk & Magazine. Awarded \$5,000 for its originality and skillful programming, Laser Chess is a two-player strategy game patterned after traditional chess—with some fascinating new twists. The original version was written in the Modula-2 language for the Atari ST. Here we have provided BASIC and machine language translations for the Amiga, Commodore 64 (and Commodore 128 in 64 mode), Apple II, and Atari XL and XE. The Amiga version requires at least 512K of memory. At least one joystick is required to play the Commodore 64 and Atari versions. The Apple II version runs on any Apple II-series computer, with either DOS 3.3 or ProDOS.

Laser Chess™, as the name implies, is a chesslike strategy game for two players. The goal is to manipulate a laser-firing piece and various reflective objects to eliminate your opponent's king. As in traditional chess, there are an infinite number of ways to accomplish this.

Refer to the special notes for your computer; then type in and save a copy of the appropriate version. Be sure to read the general game rules before you play.

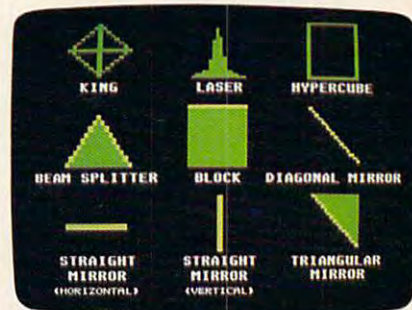
There are eight basic types of pieces in *Laser Chess*, and each has unique capabilities. Over time, you'll learn each piece's advantages and limitations. Obviously, the more you play *Laser Chess*, the more you'll understand the pieces in your arsenal, which in turn will make you a better player. So let's start with a description of the pieces.

A Geometric Army

Figure 1 shows each piece and its name. (The appearance of game pieces differs slightly in some versions; see the photos for your particular computer.) Notice that some sides of certain pieces are highlighted with a different color. This indicates a reflective surface. When a laser beam strikes a reflective surface, it bounces off without harming the piece. But if a piece is hit by a laser on a nonreflective surface, it is destroyed.

A piece can also be removed from the board if it is captured by an opposing piece. This is similar to traditional chess; to capture a piece,

Figure 1: These are the basic pieces in *Laser Chess*.



you simply move one of your own pieces onto its square.

In addition to their ability to move from square to square, pieces with reflective surfaces can also be rotated in place in 90-degree increments. This lets you orient the piece to protect it against opposing laser shots, or to set up bounce shots with your own laser.

The king is the most important piece in *Laser Chess*. When the king is eliminated, the other player wins the game. Since it has no reflective surfaces, it can be destroyed by a laser from any angle. It can also be captured by an opposing piece. The king is not totally defenseless, however. It can capture any opposing piece by moving onto its square.

But you can use it for a capture only once per turn.

The second most important piece is the *laser*. This piece is your primary offensive weapon; it's the only piece which can fire a laser shot. To take aim, you can rotate it in place at 90-degree angles. Like the king, the laser is completely vulnerable to enemy laser strikes, because it has no reflective surfaces. If you lose your laser, the game is not over, but only the most skillful (or incredibly lucky) player can overcome its loss.

Tricky Pieces

The *hypercube* is an interesting piece. It can't harm an opposing piece directly, but may very well do so indirectly. When the hypercube is moved onto another piece (even your own), that piece disappears from its original position and reappears on a randomly selected empty square. This can happen only once per turn. The hypercube can be a two-edged sword; it may relocate a piece to a vulnerable position, or it may make it possible for the piece to capture an important opposing piece on the next move. The hypercube has no reflective surfaces and cannot be rotated. It is invulnerable to laser shots, however, because it's made of transparent material—a laser beam passes right through it. Remember that.

The *beam splitter* is another tricky piece. When a laser beam strikes a splitter's vertex (the point opposite its base), the beam splits in two. The two new beams travel in opposite directions, perpendicular to the original beam's path. (See Figure 2.) When a laser shot hits one of the beam splitter's reflective surfaces, it bounces off at a 90-degree angle *without* splitting. If the beam splitter's base is hit by a laser shot, it is destroyed. The beam splitter can be rotated.

The *blocks* are fairly simple pieces. However, they may impose some complex situations. A block can capture any opposing piece by moving onto that piece's square, much like a king. But unlike a king, a block has one reflective side and can be rotated as the situation demands. Therefore, blocks can be used either offensively or defensively. A laser beam that hits the

Figure 2: As seen in this magnified view, a beam splitter's vertex reflects a laser shot in two perpendicular directions.



reflective surface of a block is deflected 180 degrees—bouncing the beam back where it came from.

A *diagonal mirror* cannot be destroyed by a laser, because both of its surfaces are reflective. Diagonal mirrors can be removed from the board only when captured by a block or a king. When a laser beam strikes a diagonal mirror, the beam is deflected 90 degrees. Diagonal mirrors can be flipped to their opposite diagonal, but cannot be rotated to face horizontally or vertically.

The *horizontal mirrors* and *vertical mirrors* (known collectively as *straight mirrors*) are also invulnerable to lasers due to their reflective surfaces. When a laser hits a straight mirror on its flat surface, the beam is deflected 180 degrees. If the laser hits a straight mirror edgewise, the beam passes straight through it. (Look closely at Figure 2; a laser beam is passing through a horizontal mirror just to the left of the red beam splitter.) Straight mirrors can be rotated to become either horizontal or vertical mirrors, but not diagonal mirrors.

The *triangular mirrors* deflect laser beams just as diagonal mirrors do, but they are vulnerable to hits on their two nonreflective sides. A triangular mirror can be rotated in 90-degree increments.

Making Moves

As in the conventional game of chess, a move in *Laser Chess* consists of moving or otherwise manipulating a game piece. The specific notes for your computer explain whether a move is made by means of a mouse pointer, a joystick, or keyboard controls. Every version uses a color change of some

Figure 3: This full-screen view of *Laser Chess* shows its 9 × 9 board grid and game controls.



sort to indicate whose turn it is; for instance, in the Amiga version, the border of the game board changes color after each turn.

The same player always moves first in *Laser Chess*. There's no particular advantage or disadvantage to moving first.

A turn consists of two moves. The number of moves remaining in a turn is indicated visually on the screen. (See Figure 3.)

Before you move or rotate a piece, you must select it. The instructions for your version explain how this is done. When a piece is selected, the appearance of the piece (or the cursor, in some versions) changes.

If you accidentally select the wrong piece, you can deselect it by the same means used to select it, as long as you're still in the same square. Deselecting is usually done after rotating a piece—more on this in a moment.

After you've selected a piece, your next decision is whether to move or rotate it. Moving a distance of one square takes one move; moving two squares takes two moves (although you can move a piece two squares in one step). Since you have only two moves per turn, the maximum distance a piece can be moved in one turn is two squares. The computer does not allow illegal moves.

Pieces can be moved forward, backward, left, or right, but not diagonally. You can effectively move a piece diagonally by using two moves—forward and right, for instance.

You cannot move a piece onto a square occupied by another piece. The only exceptions are captures

with blocks and kings, and moves of the hypercube as described above.

Rotating A Piece

The instructions for your computer's version explain how to rotate a piece. The computer does not allow you to rotate a piece that's incapable of rotation. Otherwise, the piece rotates 90 degrees (one-quarter turn) clockwise. You may continue rotating the piece to any desired position before deselecting it. Rotating a piece to face any direction takes only one move, and the move is subtracted after the piece is deselected. If you deselect the piece in its original position, no move is subtracted.

You can combine a rotation and a move in a single action (except in the Atari XL/XE version). First, select the piece. Then rotate it to the direction you wish it to face. Finally, move to any adjacent square (except a diagonal) as you would normally do. The piece moves to that square and faces in the direction you've chosen. Since rotating a piece and moving a piece each take one move, this uses up your turn.

Special Features

At the center of the 9×9 board is a special square called a *hypersquare*. It absorbs laser beams and acts like a stationary hypercube. That is, if you try to move a piece onto it, the piece disappears from its original position and reappears on a randomly selected empty square. This can happen only once per turn, however.

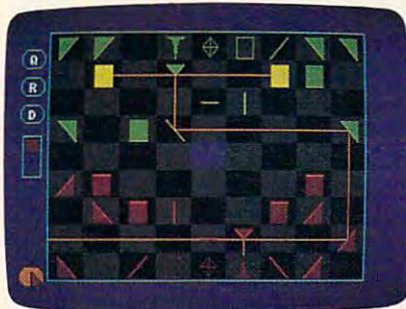
Along the board on the left side of the screen are some geometric button shapes. The button labeled Q allows you to quit playing at any time. When selected, this option requires that you confirm your decision.

The restart button (R) lets you start a new game without finishing the current game. (For instance, a player may be so hopelessly behind that he or she wants to resign.) Again, the program asks that you confirm this choice.

Firing The Laser

The last button is the laser trigger. When it's your turn, you can select this button to fire your laser. If your

Figure 4: The combination of reflective and transparent surfaces of the various pieces can result in complex bounce patterns. Here, the red laser takes advantage of the green beam splitter to destroy two blocks.



laser piece has been captured or destroyed, the laser button won't appear on the screen during your turn.

Figure 4 illustrates the general effect of firing the laser (some versions differ slightly in appearance). Firing your laser takes only one move, but can be done only once per turn. Therefore, you may want to use your first move in a turn to aim the laser, rotate a reflecting piece to set up a bounce shot, or move another piece into position.

Of course, you won't necessarily be firing the laser on every turn. Much of the strategy in *Laser Chess* involves moving and rotating your pieces to set up complex shots. It's important to realize that *any* laser hit on a piece's nonreflective or nontransparent surface will destroy that piece. You can destroy your own pieces just as easily as you can destroy your opponent's. You can even zap your own laser, particularly if you fire directly into the 180-degree reflective surface of a straight mirror or block, or if you fail to anticipate the effects of a beam splitter. Be forewarned.

Laser Chess Strategy

As in the conventional game of chess, much of the strategy in *Laser Chess* revolves around thoughtful placement of your pieces. However, the character of the game differs from that of chess in many ways. The laser, for example, can strike at long distances and in more than one direction at once. And the hy-

percube adds an extra element of uncertainty. The best strategy for any particular game depends to a great extent on the skill and personality of your opponent. However, here are some general tips you may find helpful.

Get your mirrors out early. Use them to gain the fullest potential of your laser. Try to position mirror networks on both sides of the beam splitter so you can inflict as much damage as possible.

Take advantage of the blocks. Since they "control" an area around them with their threat of capture, no other pieces can safely move within their range. Make your opponent work to displace them. Remember to rotate the reflective side of a block to the most probable direction of laser fire. If you can prevent a laser from destroying the block, your opponent will most likely have to gang up on it with two or more of his or her own blocks.

Use mirrors to protect your king. If you surround your king with straight and diagonal mirrors, there is no way it can be hit by a laser. Therefore, your opponent will have to break through your defense with blocks. (This is a pretty dirty trick, because when all of your opponent's blocks have been used, your king is almost invulnerable.) Defending your king with blocks is also a good strategy.

The hypercube should be used sparingly, since you have no idea where a relocated piece will reappear. Most players use the hypercube as a last resort—if another piece is going to be destroyed anyway, it doesn't hurt to take a chance and relocate it with the hypercube. Also, if your opponent's king is encircled with mirrors, you can march right in with your hypercube, followed by a block. This tactic may displace your opponent's defense, forcing him to evacuate the king from its mirrored fortress. Escorting the hypercube with an adjacent block prevents the opponent from attacking the hypercube with his or her king. Your opponent's only options will be to flee or be displaced.

Amiga Version

The Amiga version of *Laser Chess* (Program 1) requires 512K of mem-

When you want to talk computers..

ATARI COMPUTERS

Atari Computers

800 XL 64K Computer.....\$63.99
 65XE 64K Computer.....94.99
 130XE 132K Computer.....129.00
 520ST Monochrome System.....499.00
 520ST Color System.....749.00



Atari 1040 Color System **\$879**

Includes: 1040ST, 1 mb RAM with 3 1/2" drive built-in, 192K ROM with TOS, Basic, Logo, ST language, power supply and color monitor.

ATARI SOFTWARE

Access

Leaderboard Golf.....26.99

Accolade

Fight Night.....19.99

Activision

Music Studio.....37.99

Antic

Cad 3-D.....34.99

Batteries Included

Paperclip w/Spellpack.....39.99

Degas Elite.....48.99

Epyx

World Karate Championship.....24.99

Infocom

Zork III.....29.99

Microprose

Top Gunner.....19.99

F-15 Strike Eagle.....24.99

Origin Systems

Ultima 4.....39.99

Paradox

Wanderer (3-D).....29.99

Psygnosis

Deep Space.....34.99

Timeworks

Wordwriter ST.....51.99

VIP

Professional (GEM).....144.00

COMMODORE COMPUTERS



Commodore 128 System

Includes: CB128 Computer, CB1571 Disk Drive, CB1902 Monitor **\$759**

Commodore-64C 64K Computer189.00

Commodore-64C Sys. w/1802C.539.00

Commodore-128 128K Computer259.00

Amiga 1000 Computer.....899.00

Amiga 500.....Call for pricing

Amiga 2000.....and configuration

COMMODORE SOFTWARE

Activision

Hacker.....32.99

Broderbund

The Print Shop.....29.99

The Toy Shop.....39.99

Commodore

Textcraft w/Graphic Craft.....59.99

Assembler.....79.99

Enhancer DOS 1.2.....14.99

Discovery Software

Marauder Back-up.....32.99

Electronic Arts

Deluxe Paint.....69.99

Deluxe Print.....74.99

Instant Music.....34.99

Deluxe Video.....69.99

Infocom

Hitchhiker's Guide.....31.99

Micro Illusions

Dynamic-Cad.....349.00

Mindscape

Halley Project.....31.99

Deja Vu.....34.99

Micro Systems

Analyze Version 2.0.....119.00

Scribble.....64.99

On-Line/Comm.....49.99

Sublogic

Flight Simulator.....37.99

V.I.P.

V.I.P. Professional.....139.00

MS/DOS SYSTEMS

AT&T 6300.....from \$1299.00

Compaq.....from 1699.00

IBM-XT.....from 1169.00

IBM-AT.....from 2599.00

Leading Edge.....from 999.00

NEC Multispeed.....from 1499.00

Panasonic Business Partnerfrom 799.00

Toshiba 1100 Plus.....from 1699.00



PC-T00 20 Meg XT-Compatible **\$999**

MULTIFUNCTION CARDS

AST

Six Pak Plus PC/XT.....\$169.00

Hercules

Color Card.....159.00

Graphics Card Plus.....209.00

Fifth Generation

Logical Connection 256K.....329.00

Quadram

Expanded Quadboard.....119.00

Video 7

EGA Video Deluxe.....389.00

Zuckerboard

Color Card w/Parallel.....89.99

MS/DOS SOFTWARE

Ashton-Tate

d-Base III +409.00

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 Demo.....239.00

Microstuf

Crosstalk XVI.....89.99

P.F.S.

First Choice.....119.00

Word Perfect Corp.

Word Perfect 4.2.....209.00



COMPUTER MAIL ORDER

www.commodore.ca

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

MODEMS	DRIVES	PRINTERS
Anchor 6480 C64/128 1200 Baud.....\$119.00 VM520 ST520/1040 1200 Baud.129.00 Atari SX212 300/1200 (ST).....99.99 Commodore Amiga 1680-1200 BPS.....169.00 CBM 1670 &C-128).....99.99 Everex Evercom 1200 Baud Internal....109.00 Hayes Smartmodem 300 External.....139.00 Smartmodem 1200 External.....389.00 Practical Peripherals 1200 BPS External.....159.00 Quadram Quadmodem II 1200 Baud.....299.00 Supra MPP-1064 AD/AA C64.....69.99 1200AT 1200 Baud Atari.....139.00 U.S. Robotics 1200 Baud External.....129.00	 <p>Atari 1050 SS/DD (XL/XE) \$139</p> Atari AA354 SS/DD Disk (ST).....\$129.00 SHD204 20 Meg Hard Drive (ST)599.00 Allied Technology Apple Half-Heights.....109.00 Commodore Amiga 1010 3 1/2".....219.00 Amiga 1020 5 1/4".....189.00 1541C.....179.00 1571.....239.00 CSI 10 mb (64-128).....1099.00 Indus GT Disk Drive.....179.00 Microbotics 20 mb Hard Drive (Amiga).....1299.00 Racore PC Jr. Expansion Chassis.....299.00 Seagate 20 MB ST-225 Hard Drive Kit...379.00 Supra 20 Meg Hard Drive (XL/XE).....679.00 20 Meg Hard Drive (ST).....569.00 Xebec 20 mb (Amiga).....899.00	Atari 1020 XL/XE Plotter.....\$29.99 1025 XL/XE Dot Matrix.....119.00 XDM121 Letter Quality.....209.00 XMM801 XL/XE Dot Matrix.....199.00 XMM804ST Dot Matrix.....189.00 Citizen MSP-10 160 cps, 80-Column....299.00 Premier 35 cps Daisywheel.....499.00 C.Itoh 8510-SP 180 cps, 80-Column.....Call 310-SEP Epson/IBM 80-Column....Call  <p>Epson LX-86 120 cps Dot Matrix \$199</p> Epson 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 PrintheadCall 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 Okimate 20 Color Printer.....129.00 ML-182 120 cps, 80-column.....239.00 ML-192+ 200 cps, 80-column...369.00 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 Micronics NX-10 120 cps, 80-column.....209.00 NX-10C 120 cps, C64 Interface.219.00 NX-15 120 cps, 132-column.....369.00 Toshiba P321 216cps, 24-Pin Printhead..479.00 P341 216cps, 24-Pin Printhead..589.00
MONITORS	DISKETTES	
 <p>Amdek 410 12" TTL Monitor \$149</p> Amdek Video 310A Amber TTL.....\$139.00 Commodore Commodore 1902.....299.00 Amiga 1080 Hi-Res Color.....269.00 Magnavox 8505 RGB/Composite.....199.00 515 RGB/Composite.....289.00 NEC 12" TTL Green or Amber.....109.00 JC-1401P3A Multi-Sync.....579.00 Princeton Graphics MAX-12 12" Amber TTL.....169.00 Taxan Model 124 12" Amber.....119.00 Zenith ZVM 1220/1230.....(ea.) 99.99	Maxell MD1-M SS/DD 5 1/4".....\$9.99 MD2-DM DS/DD 5 1/4".....10.99 MF1-DDM SS/DD 3 1/2".....12.99 MF2-DDM DS/DD 3 1/2".....21.99 Sony MD1D SS/DD 5 1/4".....8.99 MD2D DS/DD 5 1/4".....10.99 MFD-100 SS/DD 3 1/2".....13.99 MFD-200 DS/DD 3 1/2".....20.99 Hewlett-Packard Calculators 28C Scientific Pro.....199.99 18C Business Consultant.....139.95 12C Slim Financial.....74.99	

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

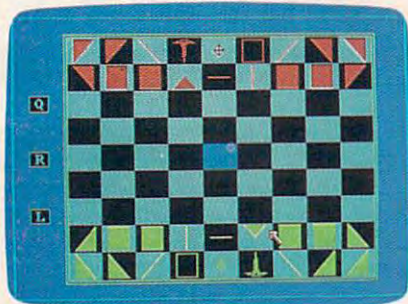
Educational, Governmental and Corporate Organizations call toll-free 1-800-221-4283

CMO. 477 East Third Street, Dept. A206, 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.

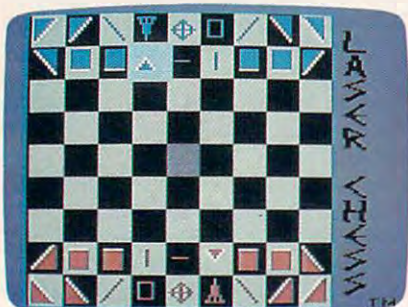
www.commodore.ca



Written in Microsoft Amiga BASIC, the Amiga version of Laser Chess duplicates almost exactly the original version of the game, which was written in the Modula-2 language for the Atari ST.



The Commodore 64 version of Laser Chess uses high-resolution graphics and sprites to good advantage.



Laser Chess for eight-bit Atari computers uses a graphics mode available only on the Atari XE and XL models.



The Apple II version of Laser Chess employs keyboard controls and runs on any Apple II-series computer.

ory and Microsoft Amiga BASIC. At the beginning of the game, you can choose between filled and unfilled playing pieces by pressing F or U, respectively. This option affects only the appearance of the pieces.

Amiga *Laser Chess* is played with the mouse, just like the original version for the Atari ST. To move a piece, position the mouse pointer over the desired piece and hold down the left mouse button. When the ghosted image of that piece appears, you can either drag the piece to a new location or rotate it by pressing a key. Release the mouse button to drop the piece in its new location.

The color of the playing-field border indicates the number of turns remaining and whose turn it is. There are three buttons to the left of the board. To select a button, move the mouse pointer over the button and press the left mouse button. The button labeled L fires the laser.

Commodore 64 Version

This version of *Laser Chess* (Program 2) requires at least one joystick. Since the program is written in machine language, it must be typed in with the "MLX" machine language entry program printed elsewhere in this issue. Here are the starting and ending addresses for MLX:

Starting address: 0801
Ending address: 1BB8

A cursor indicates your position on the board. Use the joystick to move the cursor over the piece you wish to move. (If you have only one joystick, plug it into port 2. You can simulate the second joystick by pressing the left-arrow, 1, 2, and CTRL keys to move the joystick left, right, up, and down, respectively, and pressing the space bar as a substitute for the button.)

To select a piece, hold down the fire button. To rotate a piece, move the joystick and press the button at the same time. To move a piece, move the pointer to the destination square after you have selected a piece; then press the button a second time.

Atari Version

The Atari version of *Laser Chess* (Program 3) works only on Atari XL and XE models, since it uses a

graphics mode available only on those computers. The game begins by asking whether you are using one or two joysticks.

To move a piece, move the cursor onto the piece, press the fire button, and then move to the destination square and press the button a second time. The cursor turns a darker color when a piece has been selected. To rotate a piece, move the cursor over the piece and press the button until the piece has rotated to the desired position. If you decide not to rotate the piece, keep pressing until the piece is not selected any more. This version of *Laser Chess* does not allow you to combine a rotation and a move in one action. To fire the laser, move the cursor to the laser and press the button; then press L. To quit the game, press System Reset; to quit the game and start a new game, press System Reset and enter RUN.

Apple II Version

The Apple version of *Laser Chess* runs on any Apple II computer, under either DOS 3.3 or ProDOS. This program is written in two parts. The first part (Program 4) is written in BASIC. The second part (Program 5) is written in machine language and must be entered with the "MLX" program published elsewhere in this issue. Note that you must save Program 5 with the filename LASER.ML because Program 4 attempts to load the file with that filename. Enter the Program 5 addresses as indicated here:

STARTING ADDRESS? 6000
ENDING ADDRESS? 6577

Make sure that both Program 4 and Program 5 are present on the same disk before you start the game. Run Program 4 to begin.

This version of *Laser Chess* relies on keyboard controls. Use the arrow keys to move the cursor. If you are using an Apple II+, use CTRL-J to move the cursor down and CTRL-K to move it up. Press RETURN to select and place pieces. Once a piece has been selected, you can rotate it by pressing the < or > keys. The menu in the upper left portion of the screen is used to fire the laser or exit the game. Press the ESC key to enter the menu; then move to the desired menu selection with cursor controls. Press RE-

FLIGHT!

From the sophisticated realism, detail, and intellectual stimulation of Flight Simulator...



...to the brute-force fun, thrills and excitement of Jet...



...with new adventures in Scenery Disks...



...SubLOGIC. The State of the Art in Flight.

See Your Dealer. For additional product ordering information or the name of the dealer nearest you, call (800) 637-4983.



subLOGIC
Corporation
713 Edgebrook Drive
Champaign IL 61820
(217) 359-8482 Telex: 206995

ORDER LINE: (800) 637-4983
(except in Illinois, Alaska and Hawaii)

www.sublogic.com Open 7 AM to 9 PM Central Time, CST

Lycocomputer

Marketing & Consultants

Since 1981

Complete COMMODORE

(reg. \$1009.85)

128

System

- Commodore 1571 Disk Drive
- Commodore 128 Computer
- Thompson 4120 Color Monitor w/Cable



(mouse optional)

\$715

Complete COMMODORE 64E System

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

PLUS

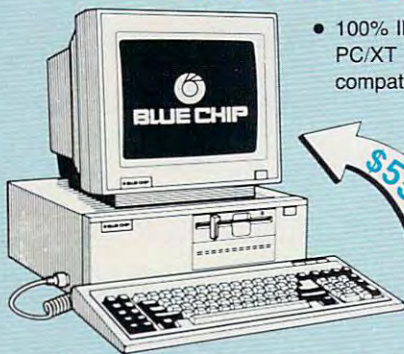
(reg. \$789)



\$509

BLUE CHIP PERSONAL COMPUTER

- 100% IBM PC/XT compatible!



(Green, amber & color monitors available)

\$559

COMMODORE HARDWARE

128 Computer	\$249	C-1700 128K RAM	\$109.95
1571 Disk Drive	\$229	1750 RAM	\$169.95
64 C Computer	\$175	Indus GT C-64 Drive	\$185
1541 C Disk Drive	\$185	GEOS	\$CALL
1902 Monitor	\$285	C-1351 Mouse	\$39
1802C Monitor	\$189	1670 Modem	\$99

ATARI 520 ST Color System

- SC 1224 Color Monitor
- SF 354 Disk Drive
- 520 Keyboard



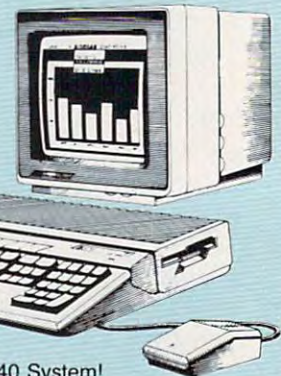
\$699

Lycocomputer offers 90-day warranty on all Atari products at no additional charge.

ATARI HARDWARE

SF 315 Disk Drive	\$219.95
SF 354 Disk Drive	\$175.95
1050 Drive (XE, XL) ..	\$139.95
SHD 204 20 MEG Drive ..	\$589
65XE	\$99
520 ST Mono	\$515
1040 Color	\$879
Indus GT Atari Drive	\$179
130XE Computer	\$119
SX 212 Modem	\$89.95

ATARI 1040 Monochrome System



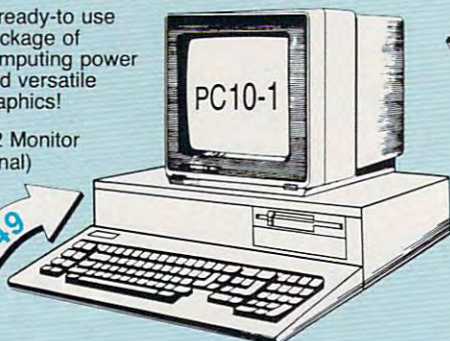
\$715

Star NP-10 Printer...\$164.95 when purchased with Atari 1040 System!

COMMODORE PC10-1

- A ready-to use package of computing power and versatile graphics!

(1902 Monitor optional)



\$749

1-800-233-8760

www.commodore.ca

Lycu 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.

The Reliable, Affordable Choice

- 120 cps Draft Mode
- 24 cps NLQ
- Word Process
- Friction Feed (2-year warranty)

Panasonic 1080i



\$195



SEIKOSHA

NP-10	\$169	SP-1200Ai ..	\$195
NX-10C	\$209	SP-1000AP IIc ..	\$179
NL-10	\$209	SP-1200AS RS232	\$195
NX-15	\$329	SL-80Ai	\$375
NB-15	\$889	MP 1300Ai	\$369
SD-10	\$259	MP 5300Ai	\$549
ND-15	\$425	BP 5420Ai	\$1075
SR-10	\$469	1300 Color Kit ..	\$119
NR-15	\$529	BP5420Ribbon.\$12.50	
NB24-15	\$729	SP1000Ribbon...\$8.50	
		1595	\$499
		3131	\$249
		3151	\$379
		1080 AP IIc	\$239

Panasonic

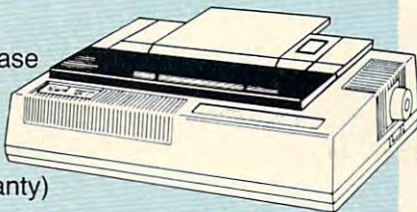
1091i	\$269
1092i	\$335
1592	\$399

Versatile and IBM Compatible



NX-10

* with purchase of printer stand limited offer (1-year warranty)

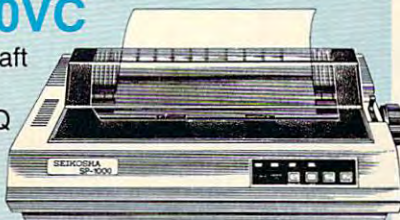


\$175

C-64/C-128 Compatible

SEIKOSHA (2-year warranty) SP-1000VC

- 100 cps Draft Mode
- 20 cps NLQ
- Direct Connect C-64/128



(retail \$299 — save \$145!)

\$154

JUKI

5510 w/color	\$435
RS232 serial board	\$55
Juki 6300	\$739
Juki 6100	\$369

LEGEND

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

SILVER REED

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

OKIDATA

Okimate 20	\$129
120 NLQ	\$209
292 w/interface	\$539
293 w/interface	\$679
182	\$245
192+	\$355
193+	\$539

DIABLO

D25	\$499
635	\$779
D-80 IF	\$1029

Toshiba

321 P/S	\$479
P341E	\$699
P351 Model II	\$1099

CITIZEN

120 D	\$179
Premiere 35	\$469
MSP-10	\$285
MSP-15	\$385
MSP-20	\$325
MSP-25	\$485
Tribute 224	\$649

EPSON

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

Monitors

Thompson:	
4120 RGB/COM	\$249
4160	\$279

Teknika:	
MJ-305	\$309
MJ-503	\$529

Zenith:	
ZVM 1220	\$89
ZVM 1230	\$89

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

Commodore:	
1902 Color	\$285
1802 C	\$189

NEC:	
Multisync	SCALL

Modems

Avatex:	
1200	\$89
1200hc	\$119
1200i	\$99
2400	\$289

Atari:	
XM-301	\$42.95
SX 212	\$89.95

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

Commodore:	
1670	\$99

US Robotics:	
Password 1200	\$189
Microlink 1200	\$139
Microlink 2400	\$219
Courier HST 9600	\$879

Compuserve	\$17.95
-------------------------	---------

Connect multiple printers with QVS Switch Boxes... starting at \$39.95!

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

.....	\$24.95	Wild. Cl. Leader Brd.	\$24.95
.....	\$New	Activision:	
.....	\$28.95	Aliens	\$24.75
.....	\$32.95	Champion. Basketball	\$22.75
.....	\$23.95	Hacker	\$18.75
.....	\$32.95	Hacker 2	\$20.75
.....	\$23.95	Labyrinth	\$22.75
.....	\$37.95	Music Studio	\$22.75
.....	\$43.95	Portal	\$26.95
.....	\$34.95	Tass Times	\$22.75
.....	\$28.95	Titanic	\$19.95
.....	\$32.95	Microprose:	
.....	\$24.95	F-15 Strike Eagle	\$22.95
.....		Gunship	\$22.95
.....	\$26.75	Kennedy Approach	\$19.95
City	\$15.95	Silent Service	\$22.95
City	\$34.95	Solo Flight	\$19.95
Leon	\$26.75	Top Gunner	\$19.95
.....		Microleague:	
.....	\$24.95	Microleag. Baseball	\$24.95
.....	\$24.95	General Manager	\$24.95
.....		Stat Disk	\$17.95
.....	\$48.95	'86 Team Disk	\$14.95
.....	\$51.75	Microleag. Wrestling	\$New
.....		Strategic Simulations:	
.....	\$24.95	Gemstone Healer	\$18.95
.....	\$24.95	Gettysburg	\$36.95
.....	\$26.95	Kampfgruppe	\$36.95
.....	\$24.95	Phantasia II	\$24.95
.....		Phantasia III	\$26.95
.....	\$24.95	Ring of Zelfin	\$24.95
.....	\$24.95	Road War 2000	\$24.95
.....	\$24.95	Shard of Spring	\$24.95
.....	\$24.95	Wizards Crown	\$24.95
.....	\$24.95	War in the S. Pacific	\$39.95
.....	\$24.95	Wargame Constr.	\$21.95
.....		Battlecruiser	\$36.95
.....		Gemstone Warrior	\$14.95
.....	\$26.95	Sublogic:	
.....	\$26.95	Baseball Stadium	\$15.95
.....	\$28.95	Flight Simulator II	\$31.95
.....	\$27.95	Jet Simulator	\$31.95
.....	\$28.95	Night Mission Pinball	\$21.95
.....		Scenery Disk	\$15.95
.....	\$174.95	Sublogic Baseball	\$31.95
.....	\$124.95	Sublogic Football	\$26.95
.....	\$74.95	Epyx:	
.....	\$18.95	Create A Calendar	\$17.95
.....	\$24.95	Destroyer	\$24.75
.....		Fastload	\$24.75
.....		Football	\$24.75
.....	\$54.95	Karate Champ	\$19.75
.....	\$79.95	Movie Monster	\$24.75
.....		Multiplan	\$24.75
.....		Sub Battle	\$23.95
.....		Winter Games	\$23.95
.....		Super Cycle	\$23.95
.....		Graphics Scrapbook	\$14.95
.....		Str. Sports Basketball	\$24.95
.....		Wild.'s Great Baseball	\$19.95
.....		Summer Games II	\$24.75
.....		Vorpal Utility Kit	\$15.75
.....		World Games	\$24.75
.....		Firebird:	
.....	\$32.75	Colossus IV	\$21.95
.....	\$22.75	Elite	\$19.95
.....	\$24.75	Frankie Goes to Italy	\$19.95
.....	\$15.75	Golden Path	\$34.95
.....	\$18.75	Guild of Thieves	\$24.95
.....	\$25.75	Pawn	\$28.95
.....	\$22.75	Sentinal	\$24.95
.....	\$12.75	Talking Teacher	\$28.95
.....	\$35.75	Tracker	\$31.95
.....		Starglider	\$24.95
.....	\$24.95	Sentry	\$24.95
.....	\$23.95	Hi Tech Expressions:	
.....	\$29.95	Heart Ware	\$6.75
.....	\$24.95	Holiday Paper	\$8.95
.....	\$13.95	Card Ware	\$6.75
.....	\$13.95	Party Ware	\$9.75
.....	\$14.95		

Unison World:	
Art Gallery 1 or 2	\$15.95
Print Master	\$22.95
Batteries Included:	
Paper Clip	\$26.95
Paper Clip II	\$44.95
Consultant 64	\$24.95
Consultant 128	\$34.95
CalKit	\$14.95
Datasoft:	
Alt. Real.: The Dungeon	\$24.95
Black Magic	\$15.95
Saracen	\$15.95
Action Soft:	
Up Periscope	\$19.95



Access:	
Triple Pack	\$14.95
Activision:	
Enchanter Trilogy	\$49.75
Game Maker	\$31.75
Leather Goddesses	\$25.75
Term Paper	\$35.75
Champ. Baseball	\$24.95
Champ. Basketball	\$26.95
Championship Golf	\$New
GFL Football	\$24.95
Portal	\$27.95
Microprose:	
Crusade in Europe	\$24.95
Decision in Desert	\$24.95
F-15 Strike Eagle	\$22.95
Silent Service	\$22.95
Microleague:	
Microleag. 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 Comp.	\$25.75
Carmen S. Diego (USA)	\$28.75
On Balance	\$62.75
Bank St. Writer +	\$44.75
Strategic Simulations:	
Battlecruiser	\$35.95
Battlegruppe	\$38.95
Colonial Conquest	\$25.95
Gettysburg	\$35.95
Phantasia II	\$25.95
Phantasia III	\$24.95
Realms of Darkness	\$24.95
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
Sublogic Baseball	\$31.95
Epyx:	
Create A Calendar	\$17.95
Destroyer	\$24.95
Karate Champ	\$19.95
Movie Monster	\$24.95
St. Sports Basketball	\$24.95
Sub Battle Simulator	\$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
Datasoft:	
Saracen	\$15.95



Access:	
Leader Board	\$24.95
10th Frame	\$24.95
Tournament #1	\$14.95
Activision:	
Borrowed Time	\$26.95
Champ. Basketball	\$26.95
Championship Golf	\$32.95
GFL Football	\$27.95
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	SCALL
Epyx:	
Apshei Trilogy	\$22.95
Rogue	\$22.95
Winter Games	\$22.95
Firebird:	
Guild of Thieves	\$27.95
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



Activision:	
Pebble Beach	\$32.95
Portal	\$28.95
Champ. Baseball	\$24.95
Champ. Basketball	\$26.95
Ballyhoo	\$25.75
Enchanter Trilogy	\$49.75
GFL Football	\$24.95
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:	
Microleag. Baseball	\$25.95
General Manager	\$25.95
Stat Disk	\$17.95
'86 Team Disk	\$14.95

Epyx:	
Apshei Trilogy	\$24.75
Create A Calendar	\$17.95
Destroyer	\$24.75
Karate Champ	\$19.75
Movie Monster	\$24.75
Rogue	\$19.75
St. Sports Basketball	\$24.95
Sub Battle Simulator	\$24.95
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
Sublogic Baseball	\$31.95
Sublogic Football	\$31.95

Broderbund:	
Ancient Art of War	\$28.95
Print Shop	\$37.75
Print Shop Compan.	\$31.75
Graphic Lib. 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
Firebird:	
Starglider	\$27.95
Guild of Thieves	\$27.95

Diskettes

5-1/4	
Maxell:	
SSDD	\$9.99
DSSD	\$12.99
Bonus:	
SSDD	\$6.99
DSSD	\$7.50
SKC:	
SSDD	\$8.50
DSSD	\$9.50
DSHD	\$16.95
Verbatim:	
SSDD	\$9.99
DSSD	\$12.99

3.5	
3M:	
SSDD	\$16.99
DSSD	\$23.99
Maxell:	
SSDD	\$15.99
DSSD	\$21.99
Verbatim:	
SSDD	\$16.99
DSSD	\$24.99
SKC:	
SSDD	\$14.99
DSSD	\$16.99

Lycos stocks joysticks by Wico, Suncom, Tac, Epyx, and more, starting as low as \$5.95!



Vickie Blaker, Customer Service Dept. Manager

Working in customer service gives me and my staff the opportunity to talk with people from all over the world who do their computer shopping by mail-order. Our loyal customers tell us that they keep coming back to Lyco because we are dedicated to customer satisfaction.

Call Lyco

Here Are Some Examples.

Giant inventory and low prices:

We hate to disappoint our customers. So, we keep a multimillion dollar inventory of all the factory-fresh merchandise you want. This means we can give you the lowest prices and the fastest delivery. And, it's also why we fill over 95% of all our customers' orders every month!

Prompt, courteous service:

When you call Lyco to place an order, you'll be in touch with some of the friendliest computer professionals in the industry. Everyone on our sales staff is very knowledgeable about the products they sell. They know that you want courteous and fast service, and that's exactly what you'll get when you call Lyco. (And for your convenience, we even have Saturday hours!)

Many companies seem to forget about customers once a sale is made. Not Lyco. Our Customer Service Department is always here to help you — from questions about the status of an order, to warranties, to product availability and price. You'll **always** find friendly service at Lyco — before, during, and after your purchase.

Fast and easy delivery:

We know that when you place an order, you'd like to receive your shipment as soon as possible. At Lyco, we don't just say it — we

do it. For instance, orders are normally shipped within 24 hours. Shipping on prepaid cash orders is free in the U.S., and there is **no deposit** required on C.O.D. orders. Air freight or UPS Blue/Red Label shipping is available, too.

TO ORDER, CALL TOLL-FREE: 1-800-233-8760 In PA: 1-717-494-1030

Hours: 9AM to 8PM, Mon.-Thurs.
9AM to 6PM, Friday – 10AM to 6PM, Saturday

Or, send orders to: Lyco Computer, Inc., P.O. Box 5088
Jersey Shore, PA 17740

For Customer Service, call 1-717-494-1670,
9AM-5PM, Mon.-Fri.

Risk-Free Policy: • full manufacturers' warranties • no sales tax outside PA
• prices show 4% cash discount; add 4% for credit cards • APO, FPO, international: add \$5 plus 3% for priority • 4-week clearance on personal checks
• we check for credit card theft • compatibility not guaranteed • return authorization required • price/availability subject to change: call to confirm

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:	
Microleag. Baseball	\$24.95
General Manager	\$24.95
Stat Disk	\$17.95
'86 Team Disk	\$14.95
Broderbund:	
Print Shop	\$25.75
Print Shop Compan.	\$22.75
Graphic Lib. 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 Antietam	\$32.95
Battlecruiser	\$35.95
Nam	\$24.95
Phantasie	\$24.95
Wargame Construc.	\$18.95
Warship	\$39.95
Wizards Crown	\$24.95
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
Firebird:	
Golden Path	\$22.95
The Pawn	\$27.95
Datasoft:	
Alt. Real.: The Dungeon.	\$24.95
Gunslinger	\$19.95
221 B Baker St.	\$19.95

GFL Football	
Game Maker	
Hacker	
Hacker 2	
Leather Goddesses	
Little People	
Moonmist	
Music Studio	
Paint Works	
Portal	
Shanghai	
Tass Times	
Bureaucracy	
Data Soft:	
Mercenary	
Mercenary - 2nd	
Alt. Reality - The	
Alt. Ritly-T. Dunge	
Microprose:	
Silent Service	
F-15 Strike Eagle	
Optimized Systems:	
Personal Pascal	
Personal Prolog	
Strategic Simulations:	
Phantasie	
Phantasie II	
Road War 2000	
Colonial Conque	
Epyx:	
Apshei Trilogy	
Sub Battle Simul	
Super Cycle	
World Games	
Wrestling	
Karate Champ	
Firebird:	
Pawn	
Starglider	
Golden Path	
Guild of Thieves	
Tracker	
VIP:	
Accounts	
VIP Professional	
VIP Professional	
Unison World:	
Art Gallery 1 or 2	
Print Master	
Zoom:	
Zoomracks	
Zoomracks II	
Microleague:	
Microleague Bas	

Order Now



Here's How



TURN to select an item from this menu.

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

Program 1: Amiga Laser Chess

Version by Tim Midkiff, Editorial Programmer

```
'Laser Chess<
'Copyright 1987 Compute! Publications, Inc.<
'All Rights Reserved<
CLEAR,25000:CLEAR ,50000&<
DEFINT a-z:DEFSNG r,g,b,mx:RANDO
MIZE TIMER:SCREEN 1,320,200,4,1<
WINDOW 3,,(0,0)-(311,186),16,1:W
INDOW OUTPUT 3:COLOR ,0<
DIM sn(8,3,1,1),es(155,1),shape(
155,87),piece(9,9),orient(9,9),c
Lr(9,9)<
DIM os(155),beamck(3,9,9),dirck(
8,3,3),bmd0(158),bmd1(22),shpt(8
)<
DIM ddrxc(1,20),ddrcy(1,20),pt(1
4)<
DIM s(255),n(255),sq(255),freq(2
0,4),shptx(8,19),shpty(8,19)<
LOCATE 1,4:PRINT CHR$(169)"1987
Compute! Publications, Inc."<
LOCATE 3,11:PRINT"All Rights Res
erved"<
LOCATE 12,9:PRINT"(F)illed or (U
)nfilled?"<
WHILE NOT(k$="F" OR k$="U"):k$=U
CASE$(INKEY$):WEND:fl=k$="F"<
PALETTE 0,.15,.05,.5:PALETTE 1,.
15,.25,.95<
FOR i=2 TO 14:PALETTE i,.15,.05,
.5:NEXT:PALETTE 15,.15,.25,.95<
ON TIMER(1) GOSUB Clock:ti=36-fl
*10<
COLOR 1,0:CLS:LOCATE 10,14:PRINT
"PLEASE WAIT"<
LOCATE 12,18:PRINT "seconds":TIM
ER ON<
FOR i=0 TO 255:s(i)=127-i:NEXT:F
OR i=0 TO 255:n(i)=127-RND*255:N
EXT:WAVE 0,s<
FOR i=0 TO 127:sq(i)=127-RND*50:
NEXT:FOR i=128 TO 255:sq(i)=-128
+RND*50:NEXT<
cop(1)=4:cop(2)=6:GOSUB InitShap
es:GOSUB InitObjects:TIMER OFF:C
LS<
RESTORE PaLetteData:FOR i=2 TO 1
4:READ r,g,b:PALETTE i,r,g,b:NEX
T<
PaLetteData: <
DATA 0,0,0,.3,.3,.3<
DATA .6,0,0,1,0,0,0,.55,0,0,.9,0
<
DATA 1,1,0,1,1,0,.6,.6,.6,1,1,1<
DATA 1,1,0,1,1,0,1,1,0<
Start:<
L(1)=1:L(2)=1:Lpx(1)=4:Lpy(1)=1:
Lpx(2)=6:Lpy(2)=9<
COLOR ,0:GOSUB DrawBoard:k=0:pL=
1<
Main:<
pL=pL XOR 3:px=5:py=5:move=2:hyc
ube=0:hysg=0:taken=0:fired=1<
LINE(40,10)-(288,186),cop(pL),b:
LINE(42,12)-(286,184),cop(pL),b<
MovePiece:<
WHILE MOUSE(0)>-1:WEND:x=MOUSE(3
):y=MOUSE(4)<
px=INT((x-17)/27):py=INT((y+6)/1
9):moves=0<
```

```
IF NOT((px>0 AND px<10) AND (py>
0 AND py<10)) THEN Options<
IF cLr(px,py)<>pL THEN MovePiece
<
piece=piece(px,py):rot=orient(px
,py)<
obindex=oi(piece,rot):spx=px:spy
=py<
IF NOT(obindex>0) THEN MovePiece
<
OBJECT.X obindex,x-14:OBJECT.Y o
bindex,y-10:OBJECT.ON obindex
<
WHILE MOUSE(0)<0<
OBJECT.X obindex,MOUSE(1)-14:OBJ
ECT.Y obindex,MOUSE(2)-10<
IF INKEY$<>" " THEN<
rot=(rot+1) AND turns(piece):j=o
bindex:obindex=oi(piece,rot)<
OBJECT.X obindex,MOUSE(1)-14:OBJ
ECT.Y obindex,MOUSE(2)-10<
OBJECT.OFF j:WAVE 0,s:SOUND 4000
,.1,255,0:OBJECT.ON obindex<
END IF <
WEND<
OBJECT.OFF obindex<
GOSUB EraseSquare<
px=INT((MOUSE(5)-17)/27):py=INT(
(MOUSE(6)+6)/19)<
GOSUB CheckMove<
GOSUB PutShape<
IF piece(px,py)=2 THEN Lpx(pL)=p
x:Lpy(pL)=py<
EndMove:IF k THEN EndGame<
move=move-moves:IF move=1 THEN L
INE(40,10)-(288,186),0,b<
IF move>0 THEN MovePiece<
GOTO Main<
<
Clock:ti=ti-1:LOCATE 12,14:PRINT
STR$(ti) " ":RETURN<
<
InitShapes:<
LINE(0,0)-(0,0),10,bf:GET(0,0)-(
0,0),pt:PUT(0,0),pt<
LINE(0,0)-(0,18),10:GET(0,0)-(0,
18),bmd0:PUT(0,0),bmd0<
LINE(0,0)-(26,0),10:GET(0,0)-(26
,0),bmd1:PUT(0,0),bmd1<
LINE(0,0)-(26,18),2,bf:GET(0,0)-
(26,18),es(0,0)<
LINE(0,0)-(26,18),3,bf:GET(0,0)-
(26,18),es(0,1)<
RESTORE LaserDir:FOR i=0 TO 3:RE
AD dirx(i),diry(i):NEXT<
LaserDir:DATA 0,-1,1,0,0,1,-1,0<
RESTORE ShapePts<
k=0:x=0:y=0:FOR i=1 TO 8:READ tu
rns(i),shpt(i)<
FOR j=0 TO shpt(i)+1:READ shptx(
i,j),shpty(i,j):NEXT<
GOSUB GetShapes:NEXT<
RESTORE ShapeRefLect<
FOR i=1 TO 8:FOR j=0 TO turns(i)
:FOR k=0 TO 3:READ dirck(i,j,k):
NEXT k,j,i<
RETURN<
ShapePts:<
DATA 1,1,-1,17,17,1,0,0<
DATA 3,6,7,17,9,1,11,17,7,17,1,1
5,17,15,11,17,9,9<
DATA 1,1,-1,9,17,9,0,0<
DATA 0,7,5,9,9,5,13,9,9,13,5,9,1
3,9,9,13,9,5,0,0<
DATA 3,6,1,2,17,2,17,17,1,17,1,2
,-1,1,17,1,9,9<
DATA 0,4,1,1,17,1,17,17,1,17,1,1
,0,0<
DATA 3,6,2,1,16,1,9,8,2,1,-1,1,-
9,9,17,1,9,4<
DATA 3,5,2,17,17,17,17,2,2,17,-1
,17,17,1,13,13<
ShapeRefLect:<
DATA 1,0,3,2,3,2,1,0,-1,-1,-1,-1
,-1,-1,-1,-1<
```

```
DATA -1,-1,-1,-1,-1,-1,-1,-1,2,1
,0,3,0,3,2,1<
DATA -1,-1,-1,-1,-1,-1,0,-1,-1,-
1,-1,1,2,-1,-1,-1<
DATA -1,3,-1,-1,0,1,2,3,-2,2,-1,
2,3,-2,3,-1<
DATA -1,0,-2,0,1,-1,1,-2,-1,0,3,
-1,-1,-1,1,0<
DATA 1,-1,-1,2,3,2,-1,-1<
<
GetShapes:<
FOR angLe=0 TO turns(i):FOR bkgd
=0 TO 1:FOR pL=1 TO 2<
co=cop(pL):PUT(0,0),es(0,bkgd),P
SET<
ON angLe+1 GOSUB rotate0,rotate9
0,rotatel80,rotate270<
sn(i,angLe,pL-1,bkgd)=k:GET(0,0)
-(26,18),shape(0,k):k=k+1<
NEXT pL,bkgd,angLe:RETURN<
<
rotate0:<
FOR j=1 TO shpt(i):IF shptx(i,j-
1)<0 THEN hue=co+1 ELSE hue=co<
LINE(ABS(shptx(i,j-1))+4+x,shpty
(i,j-1)+y)-(ABS(shptx(i,j))+4+x,
shpty(i,j)+y),hue:NEXT<
IF shptx(i,shpt(i)+1)>0 AND fl T
HEN PAINT(shptx(i,shpt(i)+1)+4+x
,shpty(i,shpt(i)+1)+y),co,co<
RETURN<
rotate90:<
FOR j=1 TO shpt(i):IF shptx(i,j-
1)<0 THEN hue=co+1 ELSE hue=co<
LINE(18-shpty(i,j-1)+4+x,ABS(shp
tx(i,j-1)+y)-(18-shpty(i,j)+4+x
,ABS(shptx(i,j)+y),hue:NEXT<
IF shptx(i,shpt(i)+1)>0 AND fl T
HEN PAINT(18-shpty(i,shpt(i)+1)+
4+x,shpty(i,shpt(i)+1)+y),co,co<
RETURN<
rotatel80:<
FOR j=1 TO shpt(i):IF shptx(i,j-
1)<0 THEN hue=co+1 ELSE hue=co<
LINE(18-ABS(shptx(i,j-1))+4+x,18
-shpty(i,j-1)+y)-(18-ABS(shptx(i
,j))+4+x,18-shpty(i,j)+y),hue:NE
XT<
IF shptx(i,shpt(i)+1)>0 AND fl T
HEN PAINT(18-shptx(i,shpt(i)+1)+
4+x,18-shpty(i,shpt(i)+1)+y),co,
co<
RETURN<
rotate270:<
FOR j=1 TO shpt(i):IF shptx(i,j-
1)<0 THEN hue=co+1 ELSE hue=co<
LINE(ABS(shpty(i,j-1))+4+x,18-ABS
(shptx(i,j)+y),hue:NEXT<
IF shptx(i,shpt(i)+1)>0 AND fl T
HEN PAINT(ABS(shpty(i,shpt(i)+1)+
4+x,18-shptx(i,shpt(i)+1)+y),co,
co<
RETURN<
InitObjects:<
k=1:si$=STRING$(26,0):POKE SADD(
si$)+11,4:POKE SADD(si$)+15,27<
POKE SADD(si$)+19,19:POKE SADD(s
i$)+21,24:POKE SADD(si$)+23,15<
FOR piece=1 TO 8:FOR angLe=0 TO
turns(piece):seLect=sn(piece,ang
Le,0,0)<
PUT(0,0),es(0,0),PSET:PUT(0,0),s
hape(0,seLect)<
oi(piece,angLe)=k:GET(0,0)-(26,1
8),os<
sd$="":FOR i=3 TO 154:sd$=sd$+MK
IS(os(i)):NEXT<
OBJECT.SHAPE k,si$+sd$:OBJECT.PL
ANES k,3,8<
k=k+1:NEXT angLe,piece:RETURN<
<
DrawBoard:<
COLOR 3,2:LINE(11,54)-STEP(16,11
),b:PAINT(12,55),2,3:LOCATE 8,3
```

```

:PRINT"Q"
LINE(11,94)-STEP(16,10),,b:PAINT
(12,95),2,3:LOCATE 13,3:PRINT"R"
LINE(11,134)-STEP(16,10),,b:PAIN
T(12,135),2,3:LOCATE 18,3:PRINT"
L"
FOR py=1 TO 9:FOR px=1 TO 9:GOSU
B EraseSquare:NEXT px,py
LINE(151,89)-(177,107),0,bf
RESTORE ShapePos:FOR py=1 TO 2:F
OR px=1 TO 9:cLr(px,py)=1:cLr(px
,py+7)=2
READ piece(px,py),orient(px,py),
orient(10-px,10-py)
piece(10-px,10-py)=piece(px,py):
NEXT px,py
FOR px=1 TO 9:FOR py=1 TO 9:IF p
iece(px,py)>0 THEN GOSUB PutShap
e
NEXT py,px
ShapePos:
DATA 8,2,0,8,2,0,1,1,1,2,2,0,4,0
,0,6,0,0
DATA 1,0,0,8,3,1,8,3,1,8,3,1,5,2
,0,5,2,0
DATA 7,2,0,3,0,0,3,1,1,5,2,0,5,2
,0,8,2,0
RETURN
PutShape:
x=px*27+16:y=py*19-6:bkgd=(px+py
+1) AND 1
PUT(x,y),shape(0,sn(piece(px,py)
,orient(px,py),cLr(px,py)-1,bkgd
)),PSET
RETURN
EraseSquare:
x=px*27+16:y=py*19-6:bkgd=(px+py
+1) AND 1:PUT(x,y),es(0,bkgd),PS
ET
RETURN
Fire:
px=Lpx(pL):py=Lpy(pL):Lx(1)=px:L
y(1)=py:dir(1)=orient(px,py)
FOR i=1 TO 3:aLive(i)=0:term(i)=
0:NEXT:aLive(1)=1
WHILE (aLive(1)=1) OR (aLive(2)=
1) OR (aLive(3)=1)
FOR i=1 TO 3:IF aLive(i)<1 THEN
AdvBeam
nLx(i)=Lx(i)+dirx(dir(i)):nLy(i)
=Ly(i)+diry(dir(i))
IF beamck(dir(i),Lx(i),Ly(i))=1
THEN EndBeam
beamck(dir(i),Lx(i),Ly(i))=1:GOT
O DrawBeam
Hit:term(i)=1:drk(i)=tdir:IF d T
HEN EndBeam
tx=px:ty=py:px=Lx(i):py=Ly(i):IF
piece(px,py)=4 THEN k=k+cLr(px,p
y)
IF piece(px,py)=2 THEN L(cLr(px,
py))=0
x=px*27+16:y=py*19-6
m=piece(px,py):shpt(0)=shpt(m):F
OR j=0 TO shpt(0)+1:shptx(0,j)=s
hptx(m,j)
shpty(0,j)=shpty(m,j):NEXT:t=i:
i=0:co=8
ON orient(px,py)+1 GOSUB rotate0
,rotate90,rotate180,rotate270
i=t:px=tx:py=ty
EndBeam:aLive(i)=-1
AdvBeam:NEXT:WEND
RETURN
DrawBeam:
x=Lx(i)*27+29:y=Ly(i)*19+3
ON dir(i) GOTO BRT,BDn,BLt
BUP:PUT(x,y-19),bmd0:GOTO CkBeam
BRT:PUT(x,y+1),bmdl:GOTO CkBeam

```

```

BDn:PUT(x+1,y),bmd0:GOTO CkBeam
BLt:PUT(x-27,y),bmdl:GOTO CkBeam
CkBeam:
IF (nLx(i)>9) OR (nLy(i)>9) OR (
nLx(i)<1) OR (nLy(i)<1) THEN End
Beam
IF nLx(i)=5 AND nLy(i)=5 THEN En
dBeam
Lx(i)=nLx(i):Ly(i)=nLy(i):IF pie
ce(nLx(i),nLy(i))=0 THEN AdvBeam
tdir=dir(i):dir(i)=dirck(piece(L
x(i),Ly(i)),orient(Lx(i),Ly(i)),
dir(i))
IF dir(i)=-1 THEN Hit
IF dir(i)>-2 THEN AdvBeam
IF aLive(2)=0 THEN j=2 ELSE j=3
aLive(j)=1:Lx(j)=Lx(i):Ly(j)=Ly(
i)
dir(i)=tdir+1 AND 3:dir(j)=tdir-
1 AND 3
GOTO AdvBeam
Laser:
k=0:d=0:GOSUB Fire
FOR i=0 TO 3:FOR x=1 TO 9:FOR y=
1 TO 9:beamck(i,x,y)=0:NEXT y,x,
i
FOR i=1 TO 3
IF term(i)=1 THEN
IF piece(Lx(i),Ly(i))>0 THEN
tx=px:ty=py:px=Lx(i):py=Ly(i):GO
SUB ExpLode:px=tx:py=ty
END IF
NEXT
TIMER OFF:d=1:GOSUB Fire
FOR i=1 TO 3
IF term(i)=1 THEN
IF piece(Lx(i),Ly(i))>0 THEN
tx=px:ty=py:px=Lx(i):py=Ly(i):pi
ece(px,py)=0:cLr(px,py)=0
GOSUB EraseSquare:px=tx:py=ty
END IF
END IF
NEXT
FOR i=0 TO 3:FOR x=1 TO 9:FOR y=
1 TO 9:beamck(i,x,y)=0:NEXT y,x,
i
RETURN
ExpLode:
FOR j=0 TO 4:vol(4-j)=(j+1)*40:N
EXT:ch=0
FOR j=0 TO 20:t=900-INT(RND*8)*1
00:FOR m=0 TO 4:freq(j,m)=t:NEXT
m,j
Lv=120:cx=px*27+29:cy=py*19+3:WA
VE 0,n:WAVE 1,n
IF dirx(drk(i))=0 THEN
FOR j=0 TO 20:ddrcy(0,j)=INT(RND
*10)*diry(drk(i))+cy
ddrcx(0,j)=cx+10-INT(RND*20)
ddrcy(1,j)=INT(RND*20)*diry(drk(
i))+cy
ddrcx(1,j)=cx+20-INT(RND*40):NEX
T
ELSE
FOR j=0 TO 20:ddrcx(0,j)=INT(RND
*10)*dirx(drk(i))+cx
ddrcy(0,j)=cy+10-INT(RND*20)
ddrcx(1,j)=INT(RND*20)*dirx(drk(
i))+cx
ddrcy(1,j)=cy+20-INT(RND*40):NEX
T
END IF
GOSUB EraseSquare
FOR j=0 TO 20:PUT(ddrcx(0,j),ddr
cy(0,j)),pt:IF (j AND 4)=4 THEN
GOSUB ExpSnd
NEXT
FOR j=0 TO 20:PUT(ddrcx(0,j),ddr
cy(0,j)),pt:PUT(ddrcx(1,j),ddrcy

```

```

(1,j)),pt
IF (j AND 4)=4 THEN GOSUB ExpSnd
NEXT:FOR j=0 TO 20:PUT(ddrcx(1,j
),ddrcy(1,j)),pt:NEXT
RETURN
ExpSnd:
ch=1-ch:FOR m=0 TO 4:SOUND freq(
j,m),.05,vol(m),ch:NEXT:RETURN
CheckMove:
dx=ABS(px-spx):dy=ABS(py-spy)
moves=dx+dy+ABS(rot<>orient(spx,
spy))
IF dx=0 AND dy=0 THEN ValidMove
IF NOT(px>0 AND px<10 AND py>0 A
ND py<10) THEN InValidMove
IF moves>move THEN InValidMove
IF moves=2 THEN
midx=(px+spx)/2:midy=(py+spy)/2
IF midx=5 AND midy=5 THEN InVali
dMove
IF dx=2 THEN IF piece(midx,py)<>
0 THEN InValidMove
IF dy=2 THEN IF piece(px,midy)<>
0 THEN InValidMove
IF dx=1 AND dy=1 THEN
IF ((piece(px,spy)<>0) OR (px=5
AND spy=5)) AND ((piece(spx,py)<
0) OR (spx=5 AND py=5)) THEN In
ValidMove
END IF
END IF
IF piece(px,py)<>0 THEN
IF piece=4 OR piece=5 THEN
IF taken THEN InValidMove
IF piece(px,py)=4 THEN k=cLr(px,
py)
IF piece(px,py)=2 THEN L(cLr(px,
py))=0
WAVE 0,n:WAVE 1,n
FOR i=255 TO 10 STEP -20:SOUND 4
00,.1,i,0:SOUND 400,.1,i,1:NEXT
taken=1:GOTO ValidMove
ELSEIF piece=6 THEN
IF hycube THEN InValidMove
hycube=1:GOTO HyperCube
ELSE
GOTO InValidMove
END IF
END IF
IF NOT(px=5 AND py=5) THEN Valid
Move
IF hysq THEN InValidMove
WHILE (px=5 AND py=5) OR piece(p
x,py)<>0
px=INT(RND*9+1):py=INT(RND*9+1)
WEND
WAVE 0,n:FOR i=250 TO 0 STEP -2:
SOUND 100+i*2,.03,i,0:NEXT:WAVE
1,n
GOSUB ValidMove:FOR i=0 TO 250 S
TEP 2:SOUND 100+500-i*2,.03,i,1:
NEXT
hysq=1:GOSUB PutShape
RETURN
HyperCube:
nx=INT(RND*9+1):ny=INT(RND*9+1)
IF (nx=5 AND ny=5) OR piece(nx,n
y)<>0 THEN HyperCube
WAVE 0,n:FOR i=250 TO 0 STEP -2:
SOUND 100+i*2,.03,i,0:NEXT:WAVE
1,n
piece(nx,ny)=piece(px,py):orient
(nx,ny)=orient(px,py):cLr(nx,ny)
=cLr(px,py)
GOSUB ValidMove:FOR i=0 TO 250 S
TEP 2:SOUND 100+500-i*2,.03,i,1:
NEXT
GOSUB PutShape:piece(spx,spy)=0:
cLr(spx,spy)=0:px=nx:py=ny
RETURN

```

```

ValidMove:4
piece(px,py)=piece:orient(px,py)
=rot:cLr(px,py)=cLr(spx,spy)4
IF dx>0 OR dy>0 THEN piece(spx,s
py)=0:cLr(spx,spy)=04
RETURN4
InValidMove:4
px=spx:py=spy:moves=0:RETURN4
4
Confirm:4
WINDOW 2,,(114,82)-(216,105),0,1
:WINDOW OUTPUT 2:PRINT"Are you s
ure?"4
COLOR 3,2:LINE(27,14)-STEP(16,10
),,b:PAINT(28,15),2,3:LOCATE 3,5
:PRINT"Y"4
LINE(59,14)-STEP(16,10),,b:PAINT
(68,15),2,3:LOCATE 3,9:PRINT"N"4
CkCon:WHILE MOUSE(0)>-1:WEND:x=M
OUSE(3):y=MOUSE(4):co=POINT(x,y)
4
IF NOT(co=2 OR co=3) THEN CkCon4
IF x>27 AND x<43 THEN4
c=14
ELSEIF x>59 AND x<75 THEN4
c=04
ELSE4
GOTO CkCon4
END IF4
WINDOW CLOSE 2:WHILE MOUSE(0)<>0
:WEND:RETURN4
4
Options:4
co=POINT(x,y):moves=04
IF NOT(co=2 OR co=3) THEN MovePi
ece4
IF y>133 AND y<145 AND fired AND
L(pl) THEN4
fired=0:PALETTE 10,1,1,0:PALETTE
15,1,1,04
WAVE 2,sq:WAVE 3,sq:ON TIMER(1)
GOSUB Lsnd:Lv=200:GOSUB Lsnd:TIM
ER ON4
GOSUB Laser:moves=14
PALETTE 10,.6,.6,.6:PALETTE 15,.
15,.25,.954
ELSEIF y>93 AND y<105 THEN4
GOSUB Confirm:IF c THEN Restart4
ELSEIF y>53 AND y<66 THEN4
GOSUB Confirm:IF c THEN SCREEN C
LOSE 1:WINDOW CLOSE 3:CLEAR,2500
0:END4
END IF4
GOTO EndMove4
4
Lsnd:SOUND 120,18.2,Lv,2:SOUND 1
21,18.2,Lv,3:RETURN4
4
Border:LINE(40,10)-(288,186),,b:
LINE(42,12)-(286,184),,b:RETURN4
4
Restart:4
COLOR ,0:CLS:FOR i=1 TO 9:FOR j=
1 TO 9:piece(i,j)=0:cLr(i,j)=0:N
EXT j,i4
GOTO Start4
4
EGOpt:4
co=POINT(x,y)4
IF co=2 OR co=3 THEN4
IF y>93 AND y<105 THEN Restart4
IF y>53 AND y<66 THEN SCREEN CLO
SE 1:WINDOW CLOSE 3:CLEAR,25000:
END4
END IF4
GOTO EndGWait4
4
EndGame:4
IF k=3 THEN4
COLOR 10,0:GOSUB Border:COLOR 11
:LOCATE 1,19:PRINT"Draw"4
ELSE4
IF k=2 THEN COLOR 4,0:GOSUB Bord
er:COLOR 5:LOCATE 1,16:PRINT"Red
";4

```

```

IF k=1 THEN COLOR 6,0:GOSUB Bord
er:COLOR 7:LOCATE 1,15:PRINT"Gre
en";4
PRINT" victory"4
END IF4
EndGWait:4
WHILE MOUSE(0)>-1:WEND:x=MOUSE(3
):y=MOUSE(4)4
px=INT((x-8)/27):py=INT((y+6)/19
):moves=04
IF NOT((px>0 AND px<10) AND (py>
0 AND py<10)) THEN EGOpt4
GOTO EndGWait4

```

Program 2: Commodore 64 Laser Chess

Version by Bill Chin, Editorial
Programmer

```

0801:0B 08 0A 00 9E 32 30 36 2E
0809:31 00 00 00 A9 00 8D 21 3B
0811:D0 8D 20 D0 A9 0F 8D 86 29
0819:02 20 C0 0A A9 78 A0 18 73
0821:20 1E AB A5 C6 F0 FC A9 36
0829:FF 8D 21 C8 20 7B 0B A9 FB
0831:09 8D 04 C8 A9 07 8D 21 DB
0839:D0 A9 00 8D 16 C8 A9 00 1C
0841:85 FD A9 C0 85 FE A9 CE 1F
0849:85 FB A9 18 85 FC A0 00 33
0851:B1 FB F0 04 C9 19 90 0C 77
0859:91 FD 20 FD 08 20 EC 08 38
0861:D0 EE F0 17 AA A9 FF 91 B2
0869:FD CA F0 07 20 EC 08 D0 4F
0871:F4 F0 08 20 FD 08 20 EC 78
0879:08 D0 D5 20 04 09 20 4C F5
0881:0B 20 0A 0E AD 16 C8 49 F5
0889:01 8D 16 C8 20 06 0B F0 EC
0891:F0 A9 07 20 91 0A A9 03 72
0899:8D 24 C8 A9 09 8D 23 C8 BA
08A1:A9 0A 8D 31 C9 8D 32 C9 80
08A9:A2 01 A9 05 20 2A 10 20 BA
08B1:15 17 20 38 0F 20 8B 0F B8
08B9:AD 0A C8 F0 08 DE 31 C9 33
08C1:F0 10 4C CB 08 A9 0A 9D 2D
08C9:31 C9 CA 10 DD A2 01 4C 07
08D1:AB 08 AD 23 C8 C9 09 D0 F1
08D9:B8 AD 24 C8 C9 03 F0 05 04
08E1:C9 02 D0 AD F0 20 04 09 DD
08E9:4C 82 08 E6 FD D0 02 E6 4E
08F1:FE A5 FD C9 E0 D0 04 A5 3F
08F9:FE C9 C4 60 E6 FB D0 02 65
0901:E6 FC 60 AD C6 17 A2 00 84
0909:9D E0 C8 E8 E0 12 D0 F8 B3
0911:A9 FF A2 12 9D E0 C8 E8 58
0919:E0 3F D0 F8 AD C7 17 A2 72
0921:3F 9D E0 C8 E8 E0 51 D0 21
0929:F8 A2 00 8E 25 C8 A0 08 DE
0931:BD C8 17 9D 3E C8 99 86 DF
0939:C8 BD D1 17 9D 47 C8 99 FF
0941:7D C8 BD F0 17 9D 8F C8 22
0949:BD 0B 18 9D D7 C8 BD F9 31
0951:17 9D 98 C8 BD 02 18 9D B9
0959:CE C8 E8 88 10 D2 A9 03 CC
0961:8D E5 CA A9 4D 8D E6 CA E0
0969:A9 FF 8D 21 C8 20 B3 0B 4D
0971:A9 08 8D 23 C8 8D 24 C8 CB
0979:20 40 0D AD 0E C8 C9 FF 4F
0981:F0 06 20 A3 0C 4C 8C 09 7F
0989:20 FC 0B CE 23 C8 10 E8 7E
0991:A9 08 8D 23 C8 CE 24 C8 F0
0999:10 DE A9 04 8D 23 C8 8D F8
09A1:24 C8 20 40 0D A9 00 8D 9C
09A9:21 C8 20 5F 0C A9 01 8D 0F
09B1:66 C8 A9 03 8D 20 C8 8D 9A
09B9:24 C8 A9 16 8D 31 C9 A9 15
09C1:01 8D 23 C8 8D 1F C8 20 43
09C9:DE 09 EE 31 C9 EE 24 C8 99
09D1:AD 24 C8 C9 06 D0 E8 EE AD
09D9:24 C8 EE 24 C8 AD 31 C9 79
09E1:8D 02 C8 8D 00 C8 AD 24 CF
09E9:C8 8D 20 C8 20 58 0D 20 F0

```

```

09F1:A3 0C A9 00 8D 21 C8 20 B0
09F9:5F 0C A9 E1 8D 22 C8 20 B8
0A01:81 0C 60 8A 48 A9 08 8D 14
0A09:23 C8 8D 24 C8 20 40 0D 29
0A11:AD 0E C8 CD 1A C8 D0 14 1F
0A19:20 E6 0B AD 21 C8 8D 22 9C
0A21:C8 20 81 0C A9 FF AC 12 4B
0A29:C8 99 E0 C8 CE 23 C8 10 55
0A31:DC A9 08 8D 23 C8 CE 24 F5
0A39:C8 10 D2 68 AA 60 A9 4E 0F
0A41:8D F9 57 A9 DC 8D 03 D0 14
0A49:A9 A8 8D 02 D0 A9 00 8D E8
0A51:10 D0 A9 06 8D 28 D0 A9 8F
0A59:02 8D 1D D0 A9 03 8D 15 0C
0A61:D0 20 E4 FF C9 00 D0 D6 49
0A69:A9 07 20 91 0A EE 28 D0 5E
0A71:CE 27 D0 20 E4 FF C9 00 8D
0A79:F0 F9 C9 4E F0 0A C9 59 3F
0A81:D0 E6 A9 01 8D 15 D0 60 BF
0A89:A9 01 8D 15 D0 A9 00 60 43
0A91:8D FA 0A 8A 48 98 48 AE F8
0A99:FA 0A BD DA 0A A8 BD E2 64
0AA1:0A 99 05 D4 A9 00 99 06 95
0AA9:D4 BD EA 0A 99 01 D4 BD CD
0AB1:F2 0A 99 04 D4 49 01 99 9C
0AB9:04 D4 68 A8 68 AA 60 A2 ED
0AC1:18 A9 00 9D 00 DA 6A 10 1F
0AC9:FA A9 0F 8D 18 D4 A9 FF E7
0AD1:8D 0F D4 A9 00 8D 12 D4 D8
0AD9:60 00 07 00 00 07 00 00 1B
0AE1:00 01 19 04 19 8A 92 84 36
0AE9:19 17 10 50 0A 01 64 03 77
0AF1:1E 10 10 80 20 80 80 80 77
0AF9:20 00 A0 A5 A2 8D EA 0A B7
0B01:A9 00 4C 91 02 FA AD 25 C8 A8
0B09:F0 3A C9 02 F0 22 C9 03 26
0B11:F0 33 A0 51 B9 3E C8 D0 BE
0B19:14 B9 E0 C8 CD C6 17 F0 F8
0B21:18 AD C6 17 8D 1A C8 20 7F
0B29:04 0A A9 01 60 88 10 E4 33
0B31:AD C6 17 8D 1A C8 20 04 C3
0B39:0A AD C7 17 8D 1A C8 20 B0
0B41:04 0A A9 01 60 20 04 09 B5
0B49:A9 00 60 AD 11 D0 09 20 19
0B51:8D 11 D0 AD 18 D0 09 08 85
0B59:29 0F 09 50 8D 18 D0 AD 0A
0B61:16 D0 09 10 8D 16 D0 AD EC
0B69:02 DD 09 03 8D 02 DD AD 27
0B71:00 DD 29 FC 09 02 8D 00 5F
0B79:DD 60 A9 00 A2 3F 9D 0C D9
0B81:53 CA 10 FA A9 FF A2 00 38
0B89:9D C0 53 9D F6 53 E8 E0 9A
0B91:06 D0 F5 A2 06 A9 80 9D 3D
0B99:C0 53 A9 01 9D C2 53 E8 B1
0BA1:E8 E8 E0 36 90 EF A2 3F AE
0BA9:BD 79 1B 9D 80 53 CA 10 31
0BB1:F7 60 A9 00 8D C3 0B A9 4C
0BB9:60 8D C4 0B AD 21 C8 A0 D0
0BC1:00 99 41 C9 C8 D0 FA EE 71
0BC9:C4 0B AE C4 0B E0 80 D0 D4
0BD1:F0 A0 00 A9 00 99 00 D8 62
0BD9:99 00 D9 99 00 DA 99 E8 19
0BE1:DA C8 D0 F1 60 A9 0B 8D 1E
0BE9:21 C8 AD 23 C8 18 6D 24 50
0BF1:C8 29 01 D0 05 A9 0F 8D 5E
0BF9:21 C8 60 20 0B D0 A9 FF C0
0C01:8D 22 C8 AC 12 C8 99 3E 71
0C09:C8 99 8F C8 99 E0 C8 A9 F5
0C11:00 8D 02 C8 20 46 0D 20 AD
0C19:25 0C 20 B3 0C 20 2F 0C 51
0C21:20 5C 0C 60 AD 2A C8 85 15
0C29:FD AD 2B C8 85 FE A0 00 07
0C31:AD 22 C8 91 FD C8 C0 18 87
0C39:D0 F9 60 AD 2A C8 85 FD 9C
0C41:AD 2B C8 85 FE AD 2C C8 3C
0C49:85 FB AD 2D C8 85 FC A0 A2
0C51:00 B1 FB 91 FD C8 C0 18 1B
0C59:D0 F7 60 20 E6 0B A0 00 8A
0C61:AD 2E C8 85 FD AD 2F C8 1B
0C69:85 FE AD 21 C8 91 FD C8 1D
0C71:91 FD C8 91 FD A0 28 91 58
0C79:FD C8 91 FD C8 91 FD 60 BD
0C81:A0 00 AD 30 C8 85 FD AD A8

```


0C89:31	C8	85	FE	AD	22	C8	91	26	0F21:4C	39	10	8D	03	C8	4C	8B	ED	11B9:4C	C3	11	B9	47	C8	C9	FF	A1
0C91:FD	C8	91	FD	C8	91	FD	A0	16	0F29:0F	AD	0A	C8	F0	03	4C	39	6D	11C1:F0	09	A9	00	60	C9	FF	F0	EE
0C99:28	91	FD	C8	91	FD	C8	91	1E	0F31:10	8D	03	C8	4	8B	0F	A9	FF	11C9:02	D0	F7	A9	01	60	20	0B	8F
0CA1:FD	60	20	3C	0C	20	B3	0C	EC	0F39:00	8D	0A	C8	8D	0B	C8	8D	40	11D1:00	C0	28	00	2D	AD	E4	CA	71
0CA9:20	50	0C	20	81	0C	20	5C	42	0F41:17	C8	8D	1B	C8	BD	00	DC	9A	11D9:D0	3E	A9	05	20	91	0A	A9	7E
0CB1:0C	60	A5	FD	18	69	40	85	E8	0F49:8D	41	C9	4E	41	C9	B0	08	37	11E1:00	8D	41	C9	EE	E4	CA	20	EC
0CB9:FD	A5	FE	69	01	85	FE	A5	72	0F51:A9	FF	8D	1B	C8	8D	0B	C8	03	11E9:DA	0C	0C	28	F0	D0	B9	3E	84
0CC1:FB	18	69	18	85	FB	A5	FC	F0	0F59:4E	41	C9	B0	08	A9	01	8D	A9	11F1:C8	C9	FF	D0	06	20	F1	0C	98
0CC9:69	00	85	FC	60	AD	1B	D4	DB	0F61:1B	C8	8D	0B	C8	4E	41	C9	6D	11F9:4C	2F	12	EE	41	C9	D0	E7	F9
0CD1:29	0F	C9	09	B0	F7	F0	F5	49	0F69:B0	08	A9	FF	8D	17	C8	8D	FE	1201:F0	16	B9	3E	C8	C9	FF	F0	9C
0CD9:60	AD	1B	D4	6D	41	C9	29	6B	0F71:0B	C8	4E	41	C9	B0	08	A9	EF	1209:25	AD	01	C8	C9	00	F0	0B	13
0CE1:7F	C9	51	B0	F4	A8	60	49	B5	0F79:01	8D	17	C8	8D	0B	C8	4E	63	1211:C9	01	F0	20	C9	06	F0	03	C5
0CE9:FF	85	02	E6	02	A5	02	60	1D	0F81:41	C9	B0	05	A9	01	8D	0A	8F	1219:A9	00	60	AD	14	C8	D0	F8	57
0CF1:8C	23	C8	A9	00	8D	24	C8	14	0F89:C8	60	AD	23	C8	18	6D	17	A4	1221:EE	14	C8	B9	3E	C8	20	EE	BA
0CF9:AD	23	C8	38	E9	09	30	09	2B	0F91:C8	8D	23	C8	AD	24	C8	18	10	1229:15	A9	06	20	91	0A	20	95	8F
0D01:8D	23	C8	EE	24	C8	4C	F9	89	0F99:6D	1B	C8	8D	24	C8	AD	24	EA	1231:17	A9	01	60	AD	E3	CA	F0	F4
0D09:0C	60	AD	24	C8	8D	35	C9	E9	0FA1:C8	30	0E	C9	09	F0	02	90	2F	1239:03	AC	19	12	20	DA	0C	B9	74
0D11:0A	0A	0A	18	6D	35	C9	6D	B6	0FA9:0D	A9	00	8D	24	C8	4C	B7	26	1241:3E	C8	C9	FF	D0	07	C0	28	3C
0D19:23	C8	8D	35	C9	8D	12	C8	6D	0FB1:0F	A9	08	8D	24	C8	AD	23	5E	1249:F0	03	4C	56	12	EE	41	C9	2E
0D21:A8	60	B9	3E	C8	8D	00	C8	08	0FB9:C8	30	0E	C9	0A	F0	02	90	4F	1251:D0	EA	4C	19	12	8C	42	C9	C4
0D29:B9	8F	C8	8D	10	C8	18	6D	37	0FC1:0D	A9	00	8D	23	C8	4C	CF	4E	1259:AE	12	C8	20	73	17	EE	E3	2E
0D31:00	C8	8D	02	C8	B9	E0	C8	07	0FC9:0F	A9	09	8D	23	C8	60	AD	7E	1261:CA	AD	23	C8	48	AD	24	C8	51
0D39:8D	0E	C8	8D	22	C8	60	20	A4	0FD1:36	C8	85	FD	AD	37	C8	85	2F	1269:48	AC	42	C9	20	F1	0C	20	C2
0D41:0B	0D	20	23	0D	AD	24	C8	8A	0FD9:FE	A9	04	8D	21	C8	4C	5F	5F	1271:40	0D	A9	05	20	91	0A	20	F9
0D49:18	69	01	8D	20	C8	AD	23	65	0FE1:0C	20	0B	0D	20	23	0D	AD	95	1279:A3	0C	68	8D	24	C8	68	8D	FA
0D51:C8	18	69	02	8D	1F	C8	AD	4B	0FE9:12	C8	CD	13	C8	D0	35	AD	CF	1281:23	C8	20	95	17	A9	01	60	88
0D59:20	C8	0A	8D	2B	C8	A9	00	9F	0FF1:10	C8	18	6D	0B	C8	29	03	F4	1289:AD	0C	C8	8D	35	C9	AD	34	D9
0D61:8D	2A	C8	AD	20	C8	4A	6E	E7	0FF9:8D	34	C9	AD	00	C8	AA	BD	36	1291:C9	CD	11	C8	F0	05	CE	35	2B
0D69:2A	C8	18	6D	2B	C8	8D	2B	67	1001:DA	17	2D	34	C9	8D	34	C9	F3	1299:C9	30	10	AD	15	C8	F0	0E	47
0D71:C8	AD	1F	C8	0A	0A	0A	8D	E5	1009:AC	12	C8	AD	34	C9	99	8F	83	12A1:CE	15	C8	CE	35	C9	30	03	AC
0D79:35	C9	0A	18	6D	35	C9	18	4F	1011:C8	20	23	0D	20	46	0D	27	12A9:4C	9C	12	A9	00	60	AD	35	0A	
0D81:6D	2A	C8	8D	2A	C8	AD	2B	C9	1019:A3	0C	A9	1E	20	2A	10	A9	98	12B1:C9	8D	0C	C8	A9	01	60	A9	E7
0D89:C8	69	00	8D	2B	C8	AD	2A	3D	1021:02	4C	91	0A	A9	01	4C	91	A3	12B9:01	8D	15	D0	4C	91	0A	AD	DB
0D91:C8	18	69	48	8D	2A	C8	AD	1C	1029:0A	18	65	A2	8D	38	10	A5	3E	12C1:24	C8	C9	02	D0	06	20	3F	A1
0D99:2B	C8	69	61	8D	2B	C8	A9	13	1031:A2	CD	38	10	D0	F9	60	00	4D	12C9:0A	F0	EC	00	C9	03	D0	10	DB
0DA1:00	8D	36	C9	AD	02	C8	0A	93	1039:AE	03	C8	EC	04	C8	F0	05	83	12D1:20	3F	0A	F0	E2	A9	00	8D	71
0DA9:2E	36	C9	0A	2E	36	C9	0A	2A	1041:FE	8E	03	C8	60	A9	00	8D	9D	12D9:0C	C8	A9	03	8D	25	C8	60	8E
0DB1:2E	36	C9	0A	2E	36	C9	8D	B5	1049:03	C8	AD	23	C8	C9	09	D0	55	12E1:AD	05	C8	D0	D2	AD	24	C8	A2
0DB9:35	C9	8D	37	C9	AD	36	C9	41	1051:03	4C	0C	12	AD	05	C8	49	9B	12E9:C9	04	D0	06	F0	00	8D	0C	E2
0DC1:8D	38	C9	0E	35	C9	2E	36	2E	1059:01	8D	05	C8	F0	4E	20	40	CB	12F1:C8	60	C9	07	F0	01	60	AD	36
0DC9:C9	AD	35	C9	18	6D	37	C9	26	1061:0D	AD	0E	C8	0F	C8	D0	CE	12F9:26	C8	D0	BB	AE	16	C8	A9	42	
0DD1:8D	35	C9	AD	36	C9	6D	38	FF	1069:39	AD	12	C8	8D	13	C8	A9	54	1301:02	8D	28	C8	8D	29	C8	A2	62
0DD9:C9	8D	36	C9	AD	35	C9	69	DE	1071:02	20	91	0A	AD	2E	C8	8D	B2	1309:02	A9	04	9D	51	C9	A9	FF	F9
0DE1:00	8D	2C	C8	AD	36	C9	69	B4	1079:36	C8	AD	2F	C8	8D	37	C8	43	1311:9D	45	C9	9D	55	C9	9D	57	CE
0DE9:C0	8D	2D	C8	A9	00	8D	35	97	1081:AD	0C	C8	8D	0D	C8	AD	10	64	1319:C9	E8	E0	06	D0	EB	A9	00	64
0DF1:C9	8D	36	C9	AD	20	C8	0A	41	1089:C8	8D	11	C8	8D	34	C9	AD	9E	1321:AA	9D	63	C9	EE	E0	60	D0	69
0DF9:2E	36	C9	0A	2E	36	C9	0A	7A	1091:23	C8	8D	06	C8	AD	24	C8	95	1329:FA	9E	26	C8	CE	0C	C8	A2	B3
0E01:2E	36	C9	0A	2E	36	C9	8D	07	1099:8D	07	C8	AD	00	C8	8D	01	75	1331:51	BD	3E	C8	C9	12	F0	06	42
0E09:35	C9	8D	37	C9	AD	36	C9	92	10A1:C8	60	A9	00	8D	05	C8	A9	2F	1339:CA	10	F6	4C	B8	12	BD	E0	D6
0E11:8D	38	C9	0E	37	C9	2E	38	91	10A9:01	4C	91	0A	A9	0A	8D	04	C4	1341:C8	CD	0F	C8	D0	F2	8A	EA	C3
0E19:C9	0E	37	C9	2E	38	C9	AD	B4	10B1:C8	AD	23	C8	AD	06	C8	D0	7B	1349:16	C8	9D	E5	CA	A8	20	F1	E9
0E21:37	C9	18	6D	35	C9	8D	2E	3F	10B9:16	AD	24	C8	CD	07	C8	D0	4E	1351:0C	AD	23	C8	8D	47	C9	AD	A4
0E29:C8	AD	38	C9	6D	36	C9	8D	1E	10C1:0E	A9	00	8D	15	C8	20	89	C1	1359:24	C8	8D	4D	C9	8D	24	C8	DF
0E31:2F	C8	AD	1F	C8	8D	35	C9	6F	10C9:12	20	86	17	4C	DF	10	20	5F	1361:20	0B	D0	B9	8F	C8	8D	53	A5
0E39:0A	18	6D	35	C9	8D	35	C9	1A	10D1:25	11	F0	42	20	89	12	F0	47	1369:C9	A9	00	8D	27	C8	A0	02	57
0E41:18	6D	2E	C8	8D	35	C9	AD	99	10D9:3D	20	AF	11	F0	38	4C	B8	65	1371:8C	28	C8	B9	51	C9	C9	04	E5
0E49:2F	C8	69	00	8D	36	C9	AD	E2	10E1:12	AE	16	C8	BD	E5	CA	10	31	1379:F0	1D	B9	45	C9	8D	23	C8	7E
0E51:35	C9	18	69	29	8D	2E	C8	B8	10E9:05	A9	01	8D	26	C8	A0	02	87	1381:B9	4B	C9	8D	24	C8	20	40	2E
0E59:AD	36	C9	69	D8	8D	2F	C8	CD	10F1:AD	0F	C8	99	BC	55	AE	0C	04	1389:0D	20	69	15	AC	28	C8	20	74
0E61:AD	35	C9	18	69	29	8D	30	97	10F9:C8	E0	F2	0F	C2	A9	00	99	56	1391:74	16	F0	03	20	0E	15	EE	18
0E69:C8	AD	36	C9	69	54	8D	31	A1	1101:E4	55	AD	0F	C8	AE	26	C8	A7	1399:28	C8	AC	28	C8	0C	29	C8	B2
0E71:C8	AD	02	C8	0A	8A	8D	32	EF	1109:F0	02	A9	00	99	AC	56	99	1F	13A1:F0	D1	90	CF	AD	27	C8	D0	30
0E79:C8	A9	20	18	6D	32	C8	8D	3D	1111:D4	56	88	10	DB	60	AD	0D	0E									

```

1451:C3 C9 18 7D 83 CA 9D C3 EE
1459:C9 C9 C7 90 05 A9 00 9D 47
1461:63 C9 60 8A 48 98 48 BD 55
1469:63 C9 4A 4A 8D 08 15 BD 2C
1471:C3 C9 4A 4A 8D 0C 15 91
1479:A9 00 8D 36 C9 AD 0C 15 BD
1481:0A 2E 36 C9 0A 2E 36 C9 DC
1489:0A 2E 36 C9 0A 2E 36 C9 E4
1491:0A 2E 36 C9 0A 2E 36 C9 EC
1499:8D 35 C9 AD 36 C9 18 6D 60
14A1:0C 15 8D 36 C9 A9 00 8D AC
14A9:38 C9 AD 0D 15 0A 2E 38 4C
14B1:C9 0A 2E 38 C9 0A 2E 38 35
14B9:C9 8D 37 C9 18 6D 35 C9 58
14C1:8D 35 C9 AD 36 C9 6D 38 FD
14C9:C9 8D 38 C9 8D 36 C9 BD 74
14D1:C3 C9 29 07 18 6D 35 C9 8E
14D9:8D 35 C9 AD 36 C9 69 60 36
14E1:8D 36 C9 8E 41 C9 BD 63 90
14E9:C9 29 03 AA AD 35 C9 85 A7
14F1:FD AD 36 C9 85 FE A0 00 51
14F9:B1 FD 5D 08 15 91 FD AE 40
1501:41 C9 68 A8 68 AA 60 C0 45
1509:30 C0 03 00 00 20 40 0D BC
1511:AC 28 C8 AD 12 C8 C9 28 FE
1519:D0 06 A9 04 99 51 C9 60 A8
1521:AD 0E C8 C9 FF D0 04 EE 95
1529:27 C8 60 AD 32 C8 85 FD BD
1531:AD 33 C8 85 FE AC 28 C8 34
1539:B9 51 C9 8D 32 C9 A8 B1 62
1541:FD 8D 31 C9 10 CE E7 26 46
1549:C8 AD 32 C9 8D 31 C9 4C 37
1551:D0 15 C9 04 D0 09 AC 28 CE
1559:C8 99 51 C9 4C D0 16 EE C6
1561:27 C8 AC 28 C8 4C BD 16 72
1569:AE 28 C8 BD 63 C9 48 BD 7A
1571:C3 C9 48 BD 23 CA 48 BD 67
1579:83 CA 48 BD 51 C9 A8 B9 89
1581:14 18 9D 23 CA 6D 39 C8 E8
1589:69 03 9D C3 C9 B9 18 18 96
1591:9D 83 CA 6D 38 C8 69 05 58
1599:9D 63 C9 B9 1C 18 8D 19 B5
15A1:C8 20 38 14 BD 63 C9 F0 80
15A9:10 20 64 14 20 FC 0A AD 68
15B1:1B D4 30 05 A9 01 20 2A B0
15B9:10 CE 19 C8 D0 E3 68 9D D3
15C1:83 CA 68 9D 23 CA 68 9D F9
15C9:C3 C9 68 9D C3 C9 60 AC DE
15D1:28 C8 EE 31 C9 20 BD 16 93
15D9:AC 29 C8 C8 C0 06 90 01 8A
15E1:60 8C 29 C8 CE 31 C9 CE EA
15E9:31 C9 4C BD 16 D0 04 EE 6F
15F1:25 C8 60 C9 12 D0 14 8A 10
15F9:48 A2 00 AD C6 17 D9 E0 F2
1601:C8 F0 01 E8 A9 FF 9D E5 EA
1609:CA 68 AA 60 20 40 0D AD D9
1611:00 C8 20 EE 15 A9 77 8D 2E
1619:22 C8 20 81 0C A9 77 AC 47
1621:12 C8 99 E0 C8 A2 00 BD 58
1629:63 C9 F0 06 E8 E0 60 D0 54
1631:F6 60 8E 41 C9 A9 04 20 F3
1639:91 0A A0 00 AE 41 C9 20 F2
1641:CE 0C 6D 38 C8 9D 63 C9 56
1649:9D 34 03 20 CE 0C 6D 39 6E
1651:C8 9D C3 C9 9D 94 03 20 C3
1659:AD 16 9D 23 CA 20 AD 16 10
1661:9D 83 CA 1D 23 CA F0 EF 7E
1669:E8 C8 E0 60 F0 04 C0 20 97
1671:D0 CD 60 B9 51 C9 AA BD E5
1679:14 18 79 45 C9 30 24 26
1681:C9 09 F0 20 B0 1E 99 45 6B
1689:C9 8D 23 C8 BD 18 18 85
1691:79 4B C9 30 0F C9 09 F0 2C
1699:0B B0 09 99 4B C9 8D 24 F2
16A1:C8 A9 01 60 A9 04 99 51 A4
16A9:C9 A9 00 60 AD 1B D4 30 DE
16B1:04 20 CE 0C 60 20 CE 0C AF
16B9:20 E8 0C 60 AD 31 C9 29 A6
16C1:03 8D 31 C9 AD 00 C8 C9 5E
16C9:0E D0 36 A2 00 BD 57 C9 7B
16D1:CD 12 C8 D0 0E BD 5D C9 91
16D9:CD 31 C9 D0 06 A9 04 99 F7
16E1:51 C9 60 E8 E0 06 D0 E5 6A
16E9:A2 00 BD 57 C9 30 06 E8 98
16F1:E0 06 D0 F6 60 AD 12 C8 40
16F9:9D 57 C9 AD 31 C9 9D 5D 28
1701:C9 AD 31 C9 99 51 C9 AD 95
1709:23 C8 99 45 C9 AD 24 C8 98
1711:99 4B C9 60 A9 07 8D 27 C9
1719:D0 AD 24 C8 0A 0A 0A C2
1721:69 48 8D 42 C9 AD 23 C8 FF
1729:C9 09 F0 35 0A 0A 8D 0A
1731:41 C9 0A 6D 41 C9 69 50 DE
1739:8D 41 C9 90 08 A9 01 8D 37
1741:10 D0 4C 4B 17 A9 00 8D D6
1749:10 D0 AD 41 C9 8D 00 D0 D2
1751:AD D2 C9 8D 01 D0 A9 01 98
1759:8D 15 D0 A9 4F 8D F8 57 42
1761:60 A9 00 8D 10 D0 A9 38 52
1769:8D 00 D0 AD 42 C9 8D 01 A8
1771:D0 60 BD 3E C8 99 3E C8 AD
1779:BD 8F C8 99 8F C8 BD E0 19
1781:C8 99 E0 C8 60 AD 36 C8 12
1789:8D 2E C8 AD 37 C8 8D 2F 25
1791:C8 4C 5C 0C 20 86 17 20 EC
1799:0B D0 AE 13 C8 20 73 17 5C
17A1:AD 23 C8 48 AD 24 C8 48 E4
17A9:AD 06 C8 8D 23 C8 AD 07 C0
17B1:C8 8D 24 C8 20 FC 0B 68 2C
17B9:8D 24 C8 68 8D 23 C8 20 02
17C1:40 D0 4C A3 0C D5 A2 0A 1E
17C9:0A 04 12 00 01 04 0A 0A 76
17D1:0A 06 06 0E 02 02 06 06 52
17D9:0A 00 00 01 01 01 01 03 2E
17E1:03 03 03 03 03 03 03 10
17E9:03 03 03 03 03 03 03 18
17F1:03 00 02 00 00 01 02 02 EB
17F9:02 02 02 00 01 00 02 02 F7
1801:03 00 00 00 00 01 02 00 BA
1809:00 01 01 01 01 00 00 00 B1
1811:00 00 00 00 01 00 FF FF 49
1819:00 01 00 10 0C 10 0C 04 47
1821:04 04 04 00 01 02 03 00 EA
1829:03 02 01 02 01 00 03 03 AC
1831:02 01 00 01 00 03 02 04 C6
1839:04 00 04 04 04 04 01 02 60
1841:04 04 04 04 03 04 04 04 69
1849:00 03 04 04 04 01 00 03 22
1851:02 04 04 01 04 04 02 04 4C
1859:00 FF 00 01 04 01 FF FF BD
1861:02 04 02 03 FF 03 04 04 1C
1869:04 04 04 04 04 04 04 99
1871:04 04 04 04 04 04 04 93 31
1879:11 43 4F 50 59 52 49 47 DF
1881:48 54 20 31 39 38 37 20 3B
1889:43 4F 4D 50 55 54 45 21 85
1891:20 50 55 42 4C 49 43 41 04
1899:54 49 4F 4E 53 0D 41 4C B2
18A1:4C 20 52 49 47 48 54 53 36
18A9:20 52 45 53 45 52 56 45 C1
18B1:44 0D 11 11 12 41 4E 59 06
18B9:20 4B 45 59 20 54 4F 20 1C
18C1:43 4F 4E 54 49 4E 55 45 E9
18C9:00 00 81 4C 0F 03 FE FE 70
18D1:FE FE FE 03 FB FB EA 04
18D9:AE 03 EF EF EF AF FE B8
18E1:FE FE 01 FE 03 95 A6 AA C7
18E9:01 AA 03 AF AF AF 01 AF 8E
18F1:05 FA FB FB FB FB 02 6C
18F9:AA 02 55 7D 7D 02 AF EF C5
1901:EF EF EF EF FB FB FB FB E7
1909:FB FA 02 7D 7D 55 02 AA FF
1911:02 EF EF EF EF AF 0C E7
1919:E7 E7 E7 E7 E7 E7 10 E7 9B
1921:E7 E7 E7 E7 E7 11 FA 07 3D
1929:AA 07 AF F5 07 55 07 5F C2
1931:0A FA FD 07 7F AF D7 FA BC
1939:10 FD 07 7F AF DF 13 FD EC
1941:FA 04 DF AF 7F 03 FD FA E7
1949:04 D7 AF 7F 10 F5 01 FA B6
1951:FA FA 03 55 EB AA AA AA 7F
1959:03 5F 01 AF AF AF FA 2D
1961:FA FA 03 AA AA AA AA 5F
1969:AA 03 AF AF AF AF 06 44
1971:FA FA FA FA 03 AA AA D2
1979:AA AA AA 03 DF DF DF 9F 0F
1981:9F FA FA FA FA 03 AA C5
1989:AA AA AA AA 03 9F 9F DF 71
1991:DF DF 06 FA FA FA FA D0
1999:03 AA AA AA AA 03 AF AD
19A1:AF AF AF AF FA FA FA 01 43
19A9:F5 03 AA AA AA EB 55 03 4A
19B1:AF AF AF 01 5F 06 F7 07 4A
19B9:F7 F6 F6 03 AA AA AA B4
19C1:AA 03 AF AF AF AF F6 8D
19C9:F6 F7 F7 03 AA AA AA B6
19D1:AA AA 03 AF AF AF AF AA
19D9:10 FD F6 DA 03 DF 6F AF 46
19E1:AF AF 01 FD F6 DA 04 6A 6D
19E9:AA AA 01 04 AF AF AF AF FC
19F1:07 F7 F9 FA FA FA 05 7F E1
19F9:9F A7 08 FA FA FA FA 04 54
1A01:A9 AA AA AA 05 7F 9F A7 C1
1A09:07 DA F6 FD 05 AA AA AA 09
1A11:6A DA 03 AF AF AF AF AF D7
1A19:08 F6 FD 06 AF AF AF DF 2A
1A21:07 FA FA FA FA 03 AA 1B
1A29:AA AA A9 A7 03 A7 9F 7F 82
1A31:02 FA FA F9 F7 04 9F 7F B2
1A39:14 FD F5 03 F7 05 55 55 FC
1A41:55 05 7F 5F 57 FA FA 06 E9
1A49:AA AA 06 AB AB 0A FE FE 7B
1A51:FE FE 04 7F 5F 57 55 08 48
1A59:FE FE FE FE FE 03 55 55 A0
1A61:57 5F 7F 03 7F 0D FA FA 5A
1A69:06 AA AA 06 AB AB F5 FD F6
1A71:06 55 55 55 D5 F7 03 57 E9
1A79:5F 7F 11 FD F5 D5 55 04 F4
1A81:BF BF BF BF BF 07 55 55 85
1A89:D5 F5 FD 03 BF BF BF BF 52
1A91:BF 0E FB F7 FB F7 FB 10 EF
1A99:EA E6 A6 66 6A AA 04 BF FD
1AA1:7F 7F BF 07 F9 FA FA 05 94
1AA9:BF BF AB 08 F9 FA FA F9 4F
1AB1:04 66 AB BF BF 04 BF 12 F1
1AB9:AA 6A 66 A6 E6 03 BF 7F 57
1AC1:7F BF 09 EA FB F7 FB F7 25
1AC9:FB 17 FE FE EA 05 6F AF 8B
1AD1:AF F9 07 99 EA FE FE 04 2C
1AD9:6F AF AF 6F 05 D5 DF DF BD
1AE1:DF DE DE DE 01 55 01 EB D2
1AE9:BE 04 57 F7 F7 BF B7 AF
1AF1:B7 DE DE DF DF DF D5 A7
1AF9:03 BE EB 02 55 01 B7 B7 D2
1B01:F7 B7 E7 F7 57 02 D5 DF EB
1B09:DE DE DE DE 01 55 01 D6
1B11:AA 02 FE AB 01 57 F7 05
1B19:B7 B7 F7 F7 DE DE DE DE A6
1B21:DE DF D5 01 FB FE FE 03 66
1B29:55 01 F7 F7 B7 B7 F7 CE
1B31:57 03 FE FE FE FE 03 98
1B39:AA AA AA AA 03 BF BF 10
1B41:BF BF BF 02 FE FE FE 50
1B49:FE 03 AA AA AA AA 03 18
1B51:BF BF BF BF BF 05 FE FA 56
1B59:EA EA EA 02 AA AA 00 2A 67
1B61:2A 00 03 BF AF AB AB AB 38
1B69:EA EA FA FE 04 2A 2A 2A 62
1B71:AA AA 03 AB AB AF BF 04 62
1B79:00 00 00 00 00 00 0C E7 AF
1B81:80 12 94 00 1E E7 00 12 B1
1B89:94 00 12 97 80 00 00 00 C9
1B91:12 64 80 12 94 80 0C 94 6E
1B99:80 04 94 80 04 63 00 00 59
1BA1:00 06 74 B9 E9 84 A5 02 32
1BA9:64 B9 C4 14 A5 04 E3 25 84
1BB1:E0 00 00 04 00 00 00 98

```

Program 3: Laser Chess For Atari XL And XE

Version by Rhett Anderson, Assistant Editor

```

80 0 ? "<CLEAR>":POKE 710,4:
CLR :DIM S*(7680),ML*(4
0),SI(26):AS=ADR(S*):S*
(1)=CHR$(0):S*(7680)=CH

```

```

R*(0):S*(2)=S*:GOSUB 95
00
BK 1 GRAPHICS 0:POKE 710,0:?"
LASER CHESS (tm)":?"
1 JOYSTICK OR 2";:INPUT
NUMJOY:IF NUMJOY<>1 AND
D NUMJOY<>2 THEN 1
LB 2 REM COPYRIGHT 1987 COMP
UTE! PUBLICATIONS, INC.
(3 SPACES)ALL RIGHTS RE
SERVED
CP 3 PLAYER=1:PMBASE=54279:R
AMTOP=106:SDMCTL=559:GR
ACTL=53277:HPOS0=53248:
PCOLR0=704:MOVE=2
CK 4 CN0=0:CN1=1:CN2=2:CN3=3
:CN4=4:CN5=5:CN9=9:CN20
=20:CN99=99:S*(192*40)=
CHR$(0):?"(CLEAR) Laser
Chess(tm)":;GOSUB 250
00
CL 5 DIM CLR(9,9),PIECE(9,9)
,ORIENT(9,9),A$(100),A(
11),TURNS(8),DIRC(8,15
),DIRX(3),DIRY(3),DIR(3
),BEAMCK$(399)
LL 6 DIM LX(3),LY(3),ALIVE(3
),NLX(3),NLY(3),TERM(3)
,DRK(3):GRAPHICS 2+16
NJ 7 A$="n Laser chess (TM)
nn(3 SPACES)COPYRIGHT 1
987n(6 SPACES)COMPUTER
PUBLICATIONS, INC.nn(3
1 rights reserved"
NL 8 BEAMCK$(399)=CHR$(0):BE
AMCK$(1,1)=CHR$(0):BEAM
CK$(2)=BEAMCK$
OI 10 FOR X=CN0 TO CN9:FOR Y
=CN0 TO CN9:PIECE(X,Y)
=CN0:LET CLR(X,Y)=CN0
DP 19 D=10-ABS(X-Y):E=X*Y*3+
11:IF E=11 THEN 22
JE 20 SOUND CN1,E,10,D:POKE
709,D
AK 21 SOUND CN2,E+CN1,10,D
LI 22 T=T+CN1:IF T>LEN(A$) T
HEN 30
FI 23 IF A$(T,T)="n" THEN PR
INT #6:GOTO 30
EH 24 PRINT #6:A$(T,T);
MH 30 NEXT Y:NEXT X
HC 50 RESTORE 62:FOR Y=CN1 T
O CN2:FOR X=CN1 TO CN9
:LET CLR(X,Y)=CN1:LET
CLR(X,Y+7)=CN2
NB 60 READ DUM1:READ DUM2:RE
AD DUM3:PIECE(X,Y)=DUM
1:PIECE(10-X,10-Y)=DUM
1
OD 61 ORIENT(X,Y)=DUM2+CN1:O
RIENT(10-X,10-Y)=DUM3+
CN1:NEXT X:NEXT Y
NJ 62 DATA 8,2,0,8,2,0,1,1,1
,2,2,0,4,0,0
OH 64 DATA 6,0,0,1,0,0,8,3,1
,8,3,1,8,3,1
NN 66 DATA 5,2,0,5,2,0,7,2,0
,3,0,0,3,1,1
LA 68 DATA 5,2,0,5,2,0,8,2,0
BI 70 FOR I=CN1 TO 8:READ DU
M:TURNS(I)=DUM:NEXT I:
DATA 1,3,1,0,3,0,3,3
EJ 80 FOR I=CN1 TO 8:FOR J=C
N0 TO TURNS(I):FOR K=C
N0 TO CN3:READ DUM:DIR
CK(I,J+K*CN4)=DUM:NEXT
K:NEXT J:NEXT I
JE 82 DATA 1,0,3,2,3,2,1,0,-
1,-1,-1,-1,-1,-1,-1,-1
JG 84 DATA -1,-1,-1,-1,-1,-1,-
1,-1,-1,2,1,0,3,0,3,2,1
HF 86 DATA -1,-1,-1,-1,-1,-1,-
1,0,-1,-1,-1,-1,1,2,-1,
-1,-1
HF 88 DATA -1,3,-1,-1,0,1,2,
3,-2,2,-1,2,3,-2,3,-1
IP 90 DATA -1,0,-2,0,1,-1,1,1,
-2,-1,0,3,-1,-1,-1,1,0
PJ 92 DATA 1,-1,-1,2,3,2,-1,
-1
IJ 95 GRAPHICS CN9
LP 100 FOR I=CN0 TO CN3:READ
DUM1,DUM2:DIRX(I)=DU
M1:DIRY(I)=DUM2:NEXT
I:DATA 0,-1,1,0,0,1,-
1,0
KD 101 VTAB=PEEK(134)+PEEK(1
35)*256:ATAB=PEEK(140
)+PEEK(141)*256
BG 102 OFFS=PEEK(88)+256*PEE
K(89)-ATAB:HI=INT(OFF
S/256):LO=OFFS-HI*256
:OFFS2=(INT(AS/1024)+
CN1)*1024-ATAB
OH 103 POKE VTAB+CN2,LO:POKE
VTAB+CN3,HI
FL 112 COLOR 6:PLOT CN0,CN0:
DRAWTO 79,CN0:COLOR 9
:PLOT CN0,CN1:DRAWTO
79,CN1
NA 113 S*(81,192*40)=S$
OD 114 COLOR CN0:PLOT 6,CN5:
DRAWTO 60,CN5:PLOT 6,
6:DRAWTO 60,6:S*(281,
7440)=S*(201)
LL 117 COLOR 6:FOR I=86 TO 1
05 STEP CN2:PLOT 31,I
:I:DRAWTO 36,I:NEXT I:C
OLOR CN9:FOR I=87 TO
105 STEP CN2:PLOT 31,
I
AL 118 DRAWTO 36,I:NEXT I
MB 119 GRAPHICS 63:POKE 710,
9:POKE 708,132:POKE 7
09,84:COLOR 1:PLOT 12
,5:DRAWTO 12,186:DRAW
TO 122,186:DRAWTO 122
,5:DRAWTO 12,5
NE 120 PLOT CN0,CN0:DRAWTO 1
59,CN0:DRAWTO 159,191
:DRAWTO CN0,191:DRAWT
O CN0,CN0
HG 121 GOSUB 21080:HI=INT(OFF
FS2/256):LO=OFFS2-HI*
256
LA 122 A=INT((AS/1024)+CN1)*
CN4:POKE PMBASE,A:MYB
ASE=256*A:POKE SDMCTL
,46:POKE GRCTL,CN3
IL 123 FOR I=CN1 TO CN3:POKE
53256+I,CN3:POKE VTA
B+CN2,LO:POKE VTAB+CN
3,HI
CP 124 POKE PCOLR0+I,8:POKE
HPOS0+I,(I-CN1)*32+62
:NEXT I:POKE 623,4:PO
KE 53254,166:POKE 532
55,158:POKE 53260,255
LA 125 FOR I=CN0 TO 89:HH=CN
0:IF INT(INT(I/10)/CN
2)=INT(I/10)/CN2 THEN
HH=CN1
AP 126 POKE MYBASE+787+I,113
:IF HH THEN POKE MYBA
SE+787+I,142
OE 127 POKE MYBASE+915+I,199
:IF HH THEN POKE MYBA
SE+915+I,56
IP 128 POKE MYBASE+403+I,0:I
F HH THEN POKE MYBASE
+403+I,224
OH 129 POKE MYBASE+659+I,28:
IF HH THEN POKE MYBAS
E+659+I,227
AF 130 NEXT I:FOR PX=CN1 TO
CN9:FOR PY=CN1 TO CN9
:GOSUB 8000:NEXT PY:N
EXT PX:PX=CN1:PY=PX:G
OSUB 8300:GOSUB 8900
DE 131 S=STICK((PLAYER-CN1)*
NUMJOY(CN2)):IF MOVE=C
N2 THEN HYCUBE=CN0:FI
RED=CN0:HYSQ=CN0:TAKE
N=1
EJ 135 IF STRIG((PLAYER-CN1)
*NUMJOY(CN2))=0 THEN 2
00
MG 136 IF SEL>CN0 AND PEEK(7
64)=CN0 THEN POKE 764
,255:GOTO 800
OF 138 IF S=15 THEN 131
MA 139 IF STICK((PLAYER-CN1)
*NUMJOY(CN2))<>15 THEN
139
OB 140 GOSUB 8400
NG 141 PX=PX+(S=7)-(S=11)
AY=PY+(S=13)-(S=14)
PC 160 PX=PX+(PX=CN0)*CN9-(P
X=10)*CN9
PH 170 PY=PY+(PY=CN0)*CN9-(P
Y=10)*CN9
JG 175 POKE 77,0
OE 180 GOSUB 8300
PD 185 IF SEL=CN1 THEN TX=PX
:TY=PY:PX=SPX:PY=SPY:
GOSUB 8600:PX=TX:PY=T
Y
GI 190 GOTO 131
MH 200 IF SEL>CN0 THEN 400
JC 210 IF CLR(PX,PY)=PLAYER
THEN SEL=1:SPX=PX:SPY
=PY
NP 215 IF STRIG((PLAYER-CN1)
*NUMJOY(CN2))=CN0 THEN
215
GK 220 GOTO 139
OG 280 COLOR PLAYER
JK 400 FPX=PX:FPY=PY:IF PX=S
PX AND PY=SPY THEN 70
0
IA 410 IF ABS(PX-SPX)+ABS(PY
-SPY)>MOVE THEN 139
HO 420 IF ABS(PX-SPX)=CN2 TH
EN IF PIECE((PX+SPX)/
CN2,PY)<>CN0 THEN 139
IC 430 IF ABS(PY-SPY)=CN2 TH
EN IF PIECE(PX,(PY+SP
Y)/CN2)<>CN0 THEN 139
PI 440 IF ABS(PY-SPY)<>CN1 O
R ABS(PX-SPX)<>CN1 TH
EN 450
CA 445 IF ((PIECE(PX,SPY)<>C
N0 OR (PX=CN5 AND SPY
=CN5)) AND (PIECE(SPX
,PY)<>CN0 OR (SPX=CN5
AND PY=CN5))) THEN 1
39
IE 450 IF PIECE(PX,PY)<>CN0
THEN IF PIECE(SPX,SPY)
<>CN4 AND PIECE(SPX,
SPY)<>CN5 AND PIECE(S
PX,SPY)<>6 THEN 139
CB 455 IF PX=CN5 AND PY=CN5
THEN 650
FM 460 IF PIECE(PX,PY)<>CN0
THEN 500
EL 470 TX=PX:TY=PY:PX=SPX:PY
=SPY:GOSUB 8200:PX=TX
:PY=TY
GE 475 FOR T=0 TO 10:SOUND 0
,100,10,10-T:NEXT T
CE 480 PIECE(PX,PY)=PIECE(SP
X,SPY):LET CLR(PX,PY)
=CLR(SPX,SPY):ORIENT(
PX,PY)=ORIENT(SPX,SPY
):GOSUB 8000
OH 490 PIECE(SPX,SPY)=CN0:LE
T CLR(SPX,SPY)=CN0
FF 491 MOVE=MOVE-ABS(FPX-SPX

```

```

) - ABS (FPY - SPY) : IF MOV
E = CN0 THEN MOVE = CN2 : P
LAYER = (PLAYER = CN1) + CN
1 : GOSUB 2800
BC 492 PX = FPX : PY = FPY : GOSUB 8
700 : GOTO 131
MH 500 IF PIECE (SPX, SPY) = CN4
OR PIECE (SPX, SPY) = CN
5 THEN IF TAKEN THEN
TAKEN = 0 : GOTO 600
EL 501 IF PIECE (SPX, SPY) = CN4
OR PIECE (SPX, SPY) = CN
5 THEN 139
MA 505 IF HYCUBE THEN IF PIE
CE (SPX, SPY) = 6 THEN 13
9
FF 510 HYCUBE = CN1
MB 514 NX = INT (RND (CN0) * CN9 + C
N1) : NY = INT (RND (CN0) * C
N9 + CN1)
BG 516 IF NX = CN5 AND NY = CN5
THEN 510
FB 520 IF PIECE (NX, NY) <> CN0
THEN 510
FI 525 FOR T = 0 TO 20 : SOUND 0
, 30 + T, T, (20 - T) / 2 : NEXT
T
KA 530 PIECE (NX, NY) = PIECE (PX
, PY) : ORIENT (NX, NY) = OR
IENT (PX, PY) : LET CLR (N
X, NY) = CLR (PX, PY)
JP 540 PIECE (PX, PY) = PIECE (SP
X, SPY) : ORIENT (PX, PY) =
ORIENT (SPX, SPY) : LET C
LR (PX, PY) = CLR (SPX, SPY
)
DE 550 PIECE (SPX, SPY) = CN0 : LE
T CLR (SPX, SPY) = CN0
MP 560 GOSUB 8200 : TX = PX : TY = P
Y : PX = SPX : PY = SPY : GOSUB
8200 : PX = TX : PY = TY
EC 565 FOR T = 0 TO 20 : SOUND 0
, 45 + (20 - T), T, (20 - T) / 2
: NEXT T
DD 570 GOSUB 8000 : TX = PX : TY = P
Y : PX = NX : PY = NY : GOSUB 8
000 : PX = TX : PY = TY : GOTO
491
LD 600 K = 0 : IF PIECE (PX, PY) = C
N4 THEN K = CLR (PX, PY)
GP 605 FOR T = 10 TO 20 : SOUND
0, 180, 6, 20 - T : NEXT T
JN 610 PIECE (PX, PY) = PIECE (SP
X, SPY) : ORIENT (PX, PY) =
ORIENT (SPX, SPY) : LET C
LR (PX, PY) = CLR (SPX, SPY
)
DC 620 PIECE (SPX, SPY) = CN0 : LE
T CLR (SPX, SPY) = CN0
MN 630 GOSUB 8200 : TX = PX : TY = P
Y : PX = SPX : PY = SPY : GOSUB
8200 : PX = TX : PY = TY
DM 640 GOSUB 8000 : IF K THEN
7000
HC 641 GOTO 491
DL 650 IF HYSQ THEN 139
MH 651 PX = INT (RND (CN0) * CN9 + C
N1) : PY = INT (RND (CN0) * C
N9 + CN1)
GJ 655 IF PIECE (PX, PY) <> CN0
THEN 651
PB 660 HYSQ = CN1 : GOTO 455
IA 700 RESTORE PIECE (PX, PY) *
10 + 9990
LI 702 FOR T = 0 TO 5 : SOUND 0,
100 + T * 5, 6, 5 - T : NEXT T
DH 705 IF STRIG ((PLAYER - CN1)
* NUMJOY = CN2) = CN0 THEN
705
JL 710 HOLD = ORIENT (PX, PY)
EH 720 READ DUMMY2 : TEMP = HOLD
JA 740 TEMP = TEMP + CN1 : IF TEMP
> DUMMY2 THEN TEMP = CN1
LF 760 GOSUB 8200 : ORIENT (PX,
PY) = TEMP : GOSUB 8000
EO 762 IF TEMP = HOLD THEN GOS
UB 8700 : GOTO 131
MI 765 IF STICK ((PLAYER - CN1)
* NUMJOY = CN2) <> 15 THEN
790
PF 766 IF STRIG ((PLAYER - CN1)
* NUMJOY = CN2) = CN1 THEN
765
HD 780 GOTO 740
BC 790 MOVE = MOVE - CN1 : IF MOVE
= CN0 THEN MOVE = CN2 : PL
AYER = (PLAYER = CN1) + CN1
: FIRED = CN0 : GOSUB 2800
PM 795 GOSUB 8700 : GOTO 131
FF 800 IF FIRED THEN 131
FN 801 IF PIECE (PX, PY) <> CN2
THEN 131
CM 802 SOUND CN0, 255, 10, 10 : S
OUND CN1, 254, 10, 10
DH 805 FIRED = CN1 : K = CN0 : DEAD =
0 : COLOR CN3 : GOSUB 200
0
BI 808 BEAMCK$ (399) = CHR$ (CN0
) : BEAMCK$ (CN1, CN1) = CH
R$ (CN0) : BEAMCK$ (CN2) =
BEAMCK$
NA 810 FOR I = CN1 TO CN3 : IF T
ERM (I) <> CN1 THEN 830
DM 820 IF PIECE (LX (I), LY (I))
> CN0 THEN TX = PX : TY = PY
: PX = LX (I) : PY = LY (I) : GO
SUB 9000 : PX = TX : PY = TY
CD 830 NEXT I
AK 850 DEAD = 1 : GOSUB 2000 : FOR
I = CN1 TO CN3 : IF TERM
(I) <> CN1 THEN 870
FJ 860 IF PIECE (LX (I), LY (I))
= CN0 THEN 870
LJ 861 TX = PX : TY = PY : PX = LX (I)
: PY = LY (I)
ND 862 IF PIECE (PX, PY) = CN4 T
HEN K = CN1
LN 865 PIECE (PX, PY) = CN0 : LET
CLR (PX, PY) = CN0 : GOSUB
8200 : PX = TX : PY = TY
CH 870 NEXT I
EH 875 BEAMCK$ (399) = CHR$ (0) :
BEAMCK$ (1, 1) = CHR$ (0) :
BEAMCK$ (2) = BEAMCK$
AC 876 SOUND CN0, CN0, CN0, CN0
: SOUND CN1, CN0, CN0, CN
0
HB 890 IF K THEN 7000
BJ 900 MOVE = MOVE - CN1 : IF MOVE
= CN0 THEN MOVE = CN2 : PL
AYER = (PLAYER = CN1) + CN1
: GOSUB 2800
PB 910 GOSUB 8700 : GOTO 131
CN 2000 LX (CN1) = PX : LY (CN1) = P
Y : DIR (CN1) = ORIENT (PX
, PY) - CN1
CG 2002 FOR I = CN1 TO CN3 : ALI
VE (I) = CN0 : TERM (I) = CN
0 : NEXT I : ALIVE (CN1) =
CN1
FI 2005 IF NOT (ALIVE (CN1) =
CN1 OR ALIVE (CN2) = CN
1 OR ALIVE (CN3) = CN1)
THEN 2100
CF 2010 FOR I = CN1 TO CN3 : IF
ALIVE (I) < CN1 THEN 20
90
GH 2020 NLX (I) = LX (I) + DIRX (DI
R (I)) : NLY (I) = LY (I) + D
IRY (DIR (I))
AL 2030 IF BEAMCK$ (LX (I) + LY (
I) * 10 + DIR (I) * 100, LX (
I) + LY (I) * 10 + DIR (I) * 1
00) = CHR$ (CN1) THEN 2
085
LJ 2035 BEAMCK$ (LX (I) + LY (I) *
10 + DIR (I) * 100, LX (I) +
LY (I) * 10 + DIR (I) * 100)
= CHR$ (CN1) : GOTO 3000
TERM (I) = CN1 : DRK (I) = T
DIR : IF DEAD THEN 208
5
CP 2081 TX = PX : TY = PY : PX = LX (I)
: PY = LY (I) : IF PIECE (P
X, PY) = 4 THEN K = K + CLR
(PX, PY)
DL 2083 LET CLR (PX, PY) = 3 : GOS
UB 8000 : PX = TX : PY = TY
HF 2085 ALIVE (I) = -1
IN 2090 NEXT I : GOTO 2005
KD 2100 RETURN
HK 2800 COLOR PLAYER : PLOT 12
, CN5 : DRAWTO 12, 186 : D
RAWTO 122, 186 : DRAWTO
122, CN5 : DRAWTO 12, C
N5
CG 2810 PLOT CN0, CN0 : DRAWTO
159, CN0 : DRAWTO 159, 1
91 : DRAWTO CN0, 191 : DR
AWTO CN0, CN0 : RETURN
X = LX (I) * 12 + 7 : Y = LY (I)
* 20 - CN5
MA 3010 ON DIR (I) GOTO 3030,
3040, 3050
HC 3020 IF Y < 20 THEN Y = 20
KM 3021 POKE 1632, X : POKE 163
3, Y - 19 : POKE 1634, Y : D
UMMY = USR (1648) : GOTO
5000
KF 3030 POKE 1632, X : POKE 163
3, X + 1 : POKE 1634, Y : D
UMMY = USR (1679) : GOTO
5000
DE 3040 IF Y > 171 THEN Y = 171
AI 3041 POKE 1632, X + 1 : POKE 1
633, Y : POKE 1634, Y + 19
: DUMMY = USR (1648) : GOT
O 5000
AF 3050 POKE 1632, X - 11 : POKE
1633, X : POKE 1634, Y + 1
: DUMMY = USR (1679) : GOT
O 5000
MH 3060 GOTO 5000
HD 5000 IF NLX (I) <> CN9 OR NLY
(I) <> CN9 OR NLX (I) < CN
1 OR NLY (I) < CN1 THEN
2085
EJ 5005 IF NLX (I) = CN5 AND NL
Y (I) = CN5 THEN 2085
MN 5010 LX (I) = NLX (I) : LY (I) = N
LY (I) : IF PIECE (NLX (I
), NLY (I)) = CN0 THEN 2
090
BF 5020 TDIR = DIR (I) : DIR (I) = D
IRCK (PIECE (LX (I), LY (
I)), ORIENT (LX (I), LY (
I)) - 1 + DIR (I) * 4)
GE 5030 IF DIR (I) = -1 THEN 20
80
GI 5040 IF DIR (I) > -2 THEN 20
90
NK 5050 J = CN3 : IF ALIVE (CN2) =
CN0 THEN J = CN2
MA 5060 ALIVE (J) = CN1 : LX (J) = L
X (I) : LY (J) = LY (I)
PL 5070 DIR (I) = TDIR + CN1 : IF D
IR (I) = 4 THEN DIR (I) =
CN0
CO 5071 DIR (J) = TDIR - CN1 : IF D
IR (J) = -1 THEN DIR (J)
= CN3
NB 5080 GOTO 2090
JB 7000 FOR T = 53248 TO 53255
: POKE T, 0 : NEXT T : FOR
T = 10 TO 70 STEP 5 : F
OR TT = 0 TO 10 - (T / 7) :
SOUND 0, T + 20 - TT, 10, 1
0 - TT

```

10 MHz IBM® XT Compatible Computer System Sale

Complete System for Only

\$599⁰⁰

COMPUTER DIRECT

22292 N. Pepper Road, Barrington, Ill. 60010

"The Computer Experts"

Call (312) 382-5050 or 382-5244 for Free Catalogs of Over 1000 Programs & Accessories

Best Service • Best Price • 1000 Programs • 500 Accessories • 15 Day Free Trial

10 MHz Super Turbo IBM® XT Compatible Computer System

BEST GUARANTEE

FREE CATALOGS

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

* Computes over two times
faster than the IBM® XT

Look at all
you get for
only \$599⁰⁰



This computer system is sold as a complete unit. NO SUBSTITUTIONS are allowed.

Run 1000's
of IBM®
software
programs
available.

15 Day Free Trial

90 Day Replacement Policy

Double 90 Day Policy
On Computer

The complete system

10 MHz Super Turbo XT Computer

- * 512K Memory
- * Single floppy disk drive
- * Parallel printer port
- * Serial printer port
- * Mouse/joystick port
- * RGB color graphics port
- * Hercules compatible monochrome port

MS DOS 3.2 & GW Basic

12" Hi-Res 35 MHz Green Screen Monitor

- (TTL & EGA compatible)
- Monitor interface cable
- Big Blue Printer
- RS 232 IBM to Big Blue cable
- 2 rolls of paper

Word First • Word Processor

Data First • Data Base

Calc First • Spreadsheet

List Price

Sale Price

10 MHz Super Turbo XT Computer	\$1295 ⁰⁰	\$499 ⁰⁰
* 512K Memory	\$99 ⁵	No extra cost
* Single floppy disk drive	\$129 ⁵	No extra cost
* Parallel printer port	\$59 ⁵	No extra cost
* Serial printer port	\$59 ⁵	No extra cost
* Mouse/joystick port	\$59 ⁵	No extra cost
* RGB color graphics port	\$99 ⁵	No extra cost
* Hercules compatible monochrome port	\$79 ⁵	No extra cost
MS DOS 3.2 & GW Basic	\$199 ⁰⁰	\$99 ⁰⁰
12" Hi-Res 35 MHz Green Screen Monitor	\$249 ⁰⁰	\$99 ⁰⁰
Monitor interface cable	\$24 ⁵	\$19 ⁵
Big Blue Printer	\$199 ⁰⁰	\$39 ⁵
RS 232 IBM to Big Blue cable	\$19 ⁵	\$9 ⁵
2 rolls of paper	\$19 ⁵	\$5 ⁵
Word First • Word Processor	\$99 ⁰⁰	\$39 ⁵
Data First • Data Base	\$99 ⁰⁰	\$39 ⁵
Calc First • Spreadsheet	\$99 ⁰⁰	\$39 ⁵
Total price when bought separately	\$2893⁵	\$892⁵

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 \$275⁰⁰
off sale prices!

Complete System only \$599

* Built-in the Super Turbo XT

IBM® is the trademark of International Business Machines Inc.

VISA - MASTERCARD - C.O.D.

Shipping, Handling & Insurance Charges and Information

VISA - MASTERCARD - C.O.D.

Add \$35.00 for shipping, handling and insurance. Illinois residents please add 6 1/2 % sales tax. Add \$70.00 for CANADA, PUERTO RICO, HAWAII, ALASKA and APO-FPO. All orders must be in U.S. dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA & PUERTO RICO. 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. Please call for C.O.D. charges.

CALL COMPUTER DIRECT AT
(312) 382-5050 or 382-5244

8am - 8pm CST Weekdays / 9am - 12 noon Saturdays



SALE •

Famous National Brand

• SAVE

NLQ 180

Hi-Speed Printer Sale

• 160 - 180 CPS • Near Letter Quality • Lifetime Warranty*



Sale \$ 199⁰⁰

Below Wholesale Cost Prices!

List \$499.95

10" Carriage

60% OFF LIST PRICE



NLQ-180 Premium Quality Printer

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 Policy •

NLQ-180 Print Samples

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



All New up Front Panel Controls

Fantastic Graphics

Easy to Use

Fantastic Price



— APPLE — ATARI — EPSON — **NLQ 180 SPECIFICATIONS** — IBM — COMMODORE — ETC. —

Print Buffer

8K bytes utility buffer

Printing Direction

Text Mode — Bi-directional
Graphic Mode — Uni-directional

Interface

Centronics Parallel Port

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 II \$44.95 Commodore \$29.95 IBM \$24.95 Laser 128 \$19.95 Macintosh \$49.95

Shipping, Handling & Insurance Charges

Add \$10.00 for shipping, handling, and insurance. Illinois residents please add 6 1/2% sales tax. Add \$20.00 for ALASKA, CANADA, HAWAII, PUERTO RICO & APO-FPO orders. All orders must be in U.S. Dollars. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA & PUERTO RICO. 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. 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

www.commodore.ca

Complete Apple® Compatible Word Processing System

• 15 Day Free Trial • 90 Day Immediate Replacement Policy •
Home • Business • School • Word Processing



Look At All You Get for only \$499.00

This computer system is sold as a complete unit, NO SUBSTITUTIONS are allowed.

	List Price	Sale Price
1 Laser 128 Apple Compatible Computer	\$499.00	\$395.00
2 Big Blue Printer	199.00	39.95
3 12" 35 MHz Hi-Resolution Monitor	229.95	99.00
4 Magic Window II wordprocessor	150.00	49.95
5 Magicalc Spreadsheet	150.00	49.95
6 Magic Memory Data Base	60.00	39.95
7 Laser 128 Big Blue Printer Interface	19.95	12.95
8 2 Rolls of Heat Transfer Paper	19.95	5.95

Comparable Apple System (\$2495.00) \$1327.85 **\$692.70** **All 8 Pieces Only \$499**

Save
Over
\$700⁰⁰
Off List Prices

Magic Window II: Word processing is easy and more efficient because of a simple-to-read menu system and editing commands. Many powerful features of professional quality programs and more. Supports virtually all major printer functions. Operates with most 80 column video cards or 40/70 column no-hardware modes.

Magicalc: Rated as the best electronic spreadsheet on the market for Apple. The speed with which Magicalc solves number problems allows you to accurately analyze decisions beforehand. Supports multiple RAM cards of most manufacturers in any combinations up to a full 512 K.

Magic Memory: File anything, any way you want to: names, addresses, important numbers, dates and notes. Your information is organized simply, easy to maintain, and available instantly. Supports all popular printers and video cards.

Apple® is the registered trademark of Apple Computers Inc.

Shipping, Handling & Insurance Charges and Information

Add \$35.00 for shipping, handling and insurance. Illinois residents please add 6 1/4 % sales tax. Add \$70.00 for CANADA, PUERTO RICO, HAWAII, ALASKA, and APO-FPO. WE DO NOT EXPORT TO OTHER COUNTRIES EXCEPT CANADA & PUERTO RICO. 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 Road, Barrington II. 60010
CALL (312) 382-5050 TO ORDER
We Love Our Customers!

```

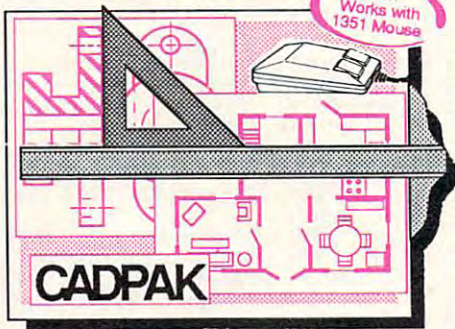
GN 7001 SOUND 1,T+21-TT,10,1
0:NEXT TT:NEXT T
LO 7002 FOR V=10 TO CN0 STEP
-0.5:SOUND CN1,254,
10,V:SOUND CN0,255,1
0,V:POKE 710,V*CN4+1
25:NEXT V
KM 7003 FOR T=1 TO 200:NEXT
T:PRINT "PLAY AGAIN";
:INPUT A$:IF A$="Y"
THEN RUN
OO 7040 GRAPHICS CN0:END
KB 8000 REM [[DRAW A PIECE O
N THE BOARD]]7030
IM 8010 PIECE=PIECE(PX,PY):L
ET CLR=CLR(PX,PY)
MM 8020 IF PIECE=CN0 THEN RE
TURN
HI 8030 ORIENT=ORIENT(PX,PY)
BI 8040 RESTORE PIECE*10+199
90
MM 8050 BRDX=12*PX+CN2:BRDY=
20*PY-14
BA 8060 READ DUMMY:COLOR CLR
HI 8070 FOR II=CN1 TO DUMMY:
READ X1,Y1,X2,Y2
AO 8090 IF X1<CN0 THEN X1=-X
1:COLOR CN3
IO 8100 IF ORIENT=CN1 THEN P
LOT BRDX+X1,BRDY+Y1*
CN2:DRAWTO BRDX+X2,B
RDY+Y2*CN2:NEXT II:R
ETURN
CA 8110 IF ORIENT=CN2 THEN P
LOT BRDX+(CN9-Y1),BR
DY+X1*CN2:DRAWTO BRD
X+(CN9-Y2),BRDY+X2*CN
2:NEXT II:RETURN
LC 8120 IF ORIENT=CN3 THEN P
LOT BRDX+(CN9-X1),BR
DY+(CN9-Y1)*CN2:DRAW
TO BRDX+(CN9-X2),BRD
Y+(CN9-Y2)*CN2:NEXT
II:RETURN
GO 8130 PLOT BRDX+Y1,BRDY+(C
N9-X1)*CN2:DRAWTO BR
DX+Y2,BRDY+(CN9-X2)*
CN2
JO 8140 NEXT II
KO 8150 RETURN
PJ 8200 REM [[ERASE BLOCK]]
ED 8210 BRDX=12*PX+CN1:BRDY=
20*PY-14:COLOR CN0
BO 8220 FOR VX=CN1 TO 6:PLOT
BRDX+VX,BRDY:DRAWTO
BRDX+VX,BRDY+19
NJ 8221 PLOT BRDX+12-VX,BRDY
:DRAWTO BRDX+12-VX,B
RDY+19:NEXT VX:RETUR
N
CC 8300 REM [[HIGHLIGHT BLOC
K]]
JL 8310 BRDX=12*PX+CN1:BRDY=
10*PY-7
MO 8320 S$(BRDY+528,BRDY+528
)=CHR$(255):S$(BRDY+
529,BRDY+539)=S$(BRD
Y+528):POKE HPOS0,BR
DX+47
PE 8325 POKE 53256,CN1
MH 8330 IF SEL=CN0 THEN POKE
PCOLR0,88+48*(PLAYE
R=CN1)
KP 8340 RETURN
MG 8400 REM [[UNHIGHLIGHT BL
OCK]]
ND 8410 BRDY=10*PY-7
CP 8420 S$(BRDY+528,BRDY+528
)=CHR$(CN0):S$(BRDY+
529,BRDY+539)=S$(BRD
Y+528)
KP 8430 RETURN
EN 8600 REM [[SELECT BLOCK]]
MK 8610 SEL=SEL+CN1:BRDX=12*
PX+CN1:BRDY=10*PY-7
BC 8620 POKE 53252,BRDX+51:F
OR T=BRDY+15 TO BRDY
+26:POKE 384+MYBASE+
T,PEEK(384+MYBASE+T)
+3:NEXT T
CG 8625 POKE PCOLR0,88+48*(P
LAYER=CN1)
LB 8630 RETURN
NH 8700 REM [[DESELECT BLOCK
]]
PN 8710 SEL=CN0:BRDY=10*SPY-
7:FOR T=BRDY+15 TO B
RDY+26:POKE 384+MYBA
SE+T,PEEK(384+MYBASE
+T)-CN3:NEXT T
EE 8720 POKE PCOLR0,88+48*(P
LAYER=CN1):RETURN
NC 8900 REM [[TURN OFF SOUND
]]
NP 8910 FOR V=10 TO CN0 STEP
-0.1:SOUND CN1,254,
10,V:SOUND CN2,255,1
0,V:POKE 710,V*CN4+1
25:NEXT V:RETURN
BA 9000 CX=12*PX+17:CY=20*PY
+4
EM 9020 FOR JJ=CN1 TO 13
JN 9030 N1=SI(JJ)+CX:N2=SI(J
J+13)+CY
LI 9040 POKE 1632,N1:POKE 16
33,N2:POKE 1634,N2+5
:DUMMY=USR(1638)
AJ 9045 SOUND CN2,N1*10,8,13
-JJ
KB 9050 NEXT JJ
KP 9060 RETURN
LO 9500 FOR JJ=1 TO 13
AH 9510 SI(JJ)=SIN(JJ)*JJ/2-
10:SI(JJ+13)=COS(JJ)
*JJ/2-10
LN 9520 NEXT JJ:RETURN
DB 10000 DATA 2,2,1,4,3,4,3,
2,1
AA 10010 DATA 4,0,0,0,0,0,0,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
DD 10020 DATA 2,3,2,1,4,1,4,
3,2
KP 10030 DATA 1,0,0,0,0
AN 10040 DATA 4,0,0,1,0,0,0,
0,2,3,0,0,0,0,4,0,0
LL 10050 DATA 1,1,2,3,4
NB 10060 DATA 4,-1,3,0,3,4,-
1,4,0,0,1,-1,1,2,0,
2,-1
BK 10070 DATA 4,0,0,2,1,2,0,
0,3,4,3,0,0,0,1,4,0
LP 20000 DATA 1,1,8,8,1
NB 20010 DATA 6,5,1,5,8,2,8,
8,8,3,8,5,1
MH 20011 DATA 7,8,5,1,2,8,3,
6,8,8,7,6
IG 20020 DATA 1,2,4,5,7,4,5
NI 20030 DATA 6,5,2,9,5,9,5,
5,8,5,8,1,5
GJ 20031 DATA 1,5,5,2,5,2,5,
8,1,5,9,5
FE 20040 DATA 14,1.5,2,1.5,8
,2,2,2,8,2.5,2
NA 20041 DATA 2.5,8,3,2,3,8,
3.5,2,3.5,8
NA 20042 DATA 4,2,4,8,4.5,2,
4.5,8,5,2,5
OA 20043 DATA 8,5.5,2,5.5,8,
6,2,6,8,6.5
OB 20044 DATA 2,6.5,8,7,2,7,
8,7.5,2,7.5
LM 20045 DATA 8,-1.5,2,7.5,2
NA 20050 DATA 4,2,2,7,2,7,2,
7,7,7,7,2,7
ED 20051 DATA 2,7,2,2
MH 20060 DATA 8,2,2,7,2
KJ 20061 DATA 2,2.5,7,2.5,3,
3,6,3
DC 20062 DATA 3,3.5,6,3.5,4,
4,5,4,4,4.5,5
MO 20063 DATA 4.5,-2,2,5,5.5
,-8,2,5,5.5
CA 20070 DATA 15,2,7,2,8,2.5
,7
NL 20071 DATA 2.5,8,3,6,3,8,
3.5,6,3.5,8
NL 20072 DATA 4,5,4,8,4.5,5,
4.5,8,5,4,5
OG 20073 DATA 8,5.5,4,5.5,8,
6,3,6,8,6.5
OF 20074 DATA 3,6.5,8,7,2,7,
8,7.5,2,7.5
CI 20075 DATA 8,8,1,8,8,8.5,
1,8,5,8
JM 20076 DATA -8,1,1,8
PL 21000 DATA 3,0,0,0,10,10,
10
MA 21010 DATA 4,0,10,5,0,10,
10,2,4
LP 21020 DATA 4,10,0,0,3,10,
6,0,10
OA 21030 DATA 5,10,0,0,6,10,
10,-10,6,0,6
KM 21040 DATA 5,0,10,0,0,10,
3,2,5,10,10
AF 21050 DATA 3,10,0,0,5,10,
10
PD 21060 DATA 6,0,0,0,10,-10
,0,10,10,-10,6,0,5
AC 21070 DATA 1,2,3,4,5,0,6,
7,4,3,3
JH 21080 RESTORE 21070:FOR I
=CN1 TO 11:READ A:A
(I)=A:NEXT I:XX=126
:YY=CN0:COLOR CN0
PF 21090 FOR I=CN1 TO 11:YY=
YY+15:IF A(I)=CN0 T
HEN NEXT I
KN 21092 FOR KK=CN0 TO CN1 S
TEP 0.5
LK 21095 RESTORE A(I)*10+209
90
DH 21100 READ DUMMY:FOR J=CN
1 TO DUMMY:READ X,Y
:FOR K=CN0 TO CN1
IF J=CN1 OR X<CN0 T
HEN PLOT 0.6*ABS(X)
+XX+KK+K,1.2*Y+YY:G
OTO 21130
LP 21120 DRAWTO 0.6*X+XX+K+K
K,1.2*Y+YY+KK/CN2
BF 21130 NEXT K:NEXT J:NEXT
KK:NEXT I
MP 21140 FOR K=1 TO 1:FOR KK
=51 TO 51
BB 21150 PLOT 135+K,130+KK:D
RAWTO 138+K,130+KK:
PLOT 136+K,130+KK:D
RAWTO 136+K,135+KK
DJ 21160 PLOT 140+K,135+KK:D
RAWTO 140+K,130+KK:
DRAWTO 142+K,134+KK
:DRAWTO 144+K,130+K
K:DRAWTO 144+K,135+
KK
JE 21170 NEXT KK:NEXT K
NM 21180 RETURN
LO 25000 DATA 6BCE6106AD6006
8D0006EE6106
EH 25010 DATA AD61068D010620
AE06AD6106
NF 25020 DATA CD6206D0E6606B
CE6006EE6006
DN 25030 DATA AD60068D0006AD
62068D0106
FN 25040 DATA 20AE06AD6006CD
6106D0E6606
LE 25045 DATA A9008D0406AD00
064A4A8D0306

```




Hop to it! check out this Great Software

Cadpak

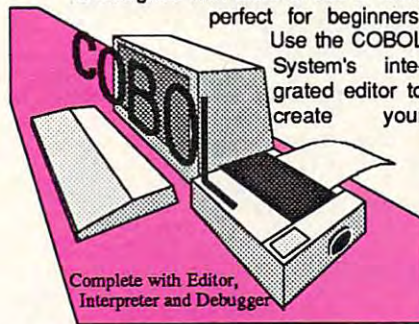


New!
Works with
1351 Mouse

Design pictures and graphics quickly and precisely. Unlike other drawing programs, you can produce exact scaled output on your printer. Design in the units of your drawing problem (feet, miles, meters, etc.) and send hardcopy to most printers. Uses either the keyboard, lightpen or 1351 mouse. Two separate work screens—transfer artwork from one screen to the other. Place text in four sizes anywhere in the picture—three extra fonts included: Old English, 3-D and Tech. "Try Again" allows you to undo mistakes. Draw solid or dashed lines, circles, ellipses at any angle, rays and boxes. Design fill patterns, fonts and objects. **Cadpak** is the full-featured design and graphics package for your computer. for C-64 \$39.95 for C-128 \$59.95

COBOL

COBOL is the most widely used commercial programming language today. COBOL is a language that is common to many computers. Most computers equipped with a COBOL system can process any COBOL program with only minor revisions. Now you can learn the COBOL language using your Commodore. COBOL is easy to learn because of its English-like syntax. COBOL is designed with ease of use in mind perfect for beginners.



Complete with Editor,
Interpreter and Debugger

Use the COBOL System's integrated editor to create your COBOL source. Then the compiler checks your program's syntax and immediately converts it into an executable form. Includes sample programs and exercises to make learning COBOL even easier for the novice or experienced programmer. for C-64 \$39.95 for C-128 \$39.95

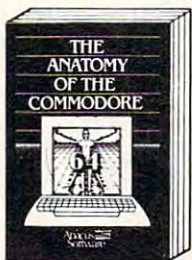
Super Pascal

Your complete system for developing applications in Pascal. A complete implementation of standard Pascal (Jensen and Wirth). C-64 version has a high-speed DOS (3X) for quick and efficient use. The extensive editor (source included) contains added features: append, search and replace. Includes assembler for any of your machine code requirements. Used in hundreds of schools to teach programming and also used for serious development projects. But it can be used for more than just learning Pascal, use it for serious programming. With complete graphic library (source included) in machine language for super-fast execution. Want to learn Pascal or develop software using the best tool? **Super Pascal** is your first choice. for C-64 \$59.95 for C-128 \$59.95

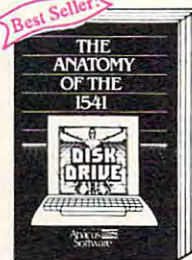
Super Pascal

Learn the world's second most widely used language!

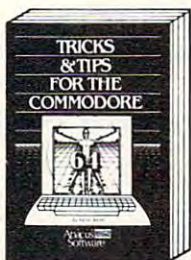
... and SUPER BOOKS!



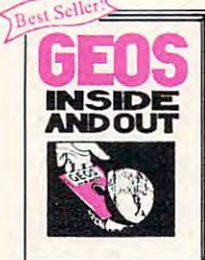
Anatomy of the C-64
Insider's guide to '64 internals. Graphics, sound, I/O, kernal, memory maps, and much more. Complete commented ROM listings. 300pp \$19.95



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



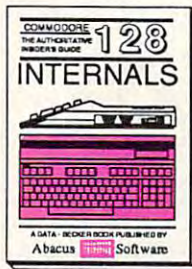
Tricks & Tips for the C-64
Collection of easy-to-use techniques: advanced graphics, improved data input, CP/M, enhanced BASIC, data handling and more. 275pp \$19.95



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 font editor, machine language monitor, quick backup, more. \$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. C6
Grand Rapids, MI 49510
Telex 709-101 • Fax 616/241-5021
Phone 616/241-5510

Commodore 64 and Commodore 128 are trademarks of Commodore Ltd.

```

CN 25050 DATA A02718AD01066D
03068D0306
ND 25060 DATA AD040669008D04
068810EB
IG 20020 DATA 1,2,4.5,7,4.5
NI 20030 DATA 6,5,2,9,5,9,5,
5,8,5,8,1,5
GJ 20031 DATA 1,5,5,2,5,2,5,
8,1,5,9,5
FE 20040 DATA 14,1.5,2,1.5,8
,2,2,2,8,2.5,2
NA 20041 DATA 2.5,8,3,2,3,8,
3.5,2,3.5,8
NA 20042 DATA 4,2,4,8,4.5,2,
4.5,8,5,2,5
DA 20043 DATA 8,5.5,2,5.5,8,
6,2,6,8,6.5
DB 20044 DATA 2,6.5,8,7,2,7,
8,7.5,2,7.5
LM 20045 DATA 8,-1.5,2,7.5,2
NA 20050 DATA 4,2,2,7,2,7,2,
7,7,7,2,7
BD 20051 DATA 2,7,2,2
MH 20060 DATA 8,2,2,7,2
KJ 20061 DATA 2,2.5,7,2.5,3,
3,6,3
DC 20062 DATA 3,3.5,6,3.5,4,
4,5,4,4,4.5,5
MD 20063 DATA 4.5,-2,2,5,5.5
,-8,2,5,5.5
CA 20070 DATA 15,2,7,2,8,2.5
,7
NL 20071 DATA 2.5,8,3,6,3,8,
3.5,6,3,5,8
NL 20072 DATA 4,5,4,8,4.5,5,
4.5,8,5,4,5
OG 20073 DATA 8,5.5,4,5.5,8,
6,3,6,8,6.5
OF 20074 DATA 3,6.5,8,7,2,7,
8,7.5,2,7.5
CI 20075 DATA 8,8,1,8,8,8.5,
1,8.5,8
JM 20076 DATA -8,1,1,8
PL 21000 DATA 3,0,0,0,10,10,
10
MA 21010 DATA 4,0,10,5,0,10,
10,2,4
LP 21020 DATA 4,10,0,0,3,10,
6,0,10
OA 21030 DATA 5,10,0,0,6,10,
10,-10,6,0,6
KM 21040 DATA 5,0,10,0,0,10,
3,2,5,10,10
AF 21050 DATA 3,10,0,0,5,10,
10
PD 21060 DATA 6,0,0,0,10,-10
,0,10,10,-10,6,0,5
GM 25070 DATA A558186D030685
D0A559
CJ 25080 DATA 6D040685D1A000
AD00062903
NJ 25090 DATA AABDF10651D091
D060C0300C03
JI 25100 DATA ENDOFDATA
DL 26000 RESTORE 25000:MEM=1
647
BD 26010 READ ML$: IF ML$="EN
DOFDATA" THEN RETUR
N
LI 26020 FOR I=CN1 TO LEN(ML
$) STEP CN2
DD 26030 MEM=MEM+CN1
P6 26040 H=ASC(ML$(I,I))-48:
IF H>9 THEN H=H-7
DB 26050 L=ASC(ML$(I+CN1,I+C
N1))-48:IF L>9 THEN
L=L-7
LB 26060 POKE MEM,H*16+L:NEX
T I:GOTO 26010

```

Program 4: Apple II Laser Chess

Version by Tim Victor, Editorial Programmer

```

BD 100 GOTO 510
99 110 GOSUB 130: IF I = SP THEN
330
15 120 RETURN
3E 130 Y = Y(I):X = X(I): IF BD(
I) = -1 THEN 200
9E 140 GOSUB 180: IF X < > 2 * I
NT (X / 2) THEN C = 1 - C
17 150 C = C + 2 * (I - 2 * INT
(I / 2))
59 160 CALL 25306,SN(P) + D,C,X,
Y
IF 170 RETURN
A3 180 C = C(BD(I)):P = P(BD(I))
:D = D(BD(I)): RETURN
23 190 RETURN
E2 200 CALL 25309,X,Y
77 210 IF X = 4 AND Y = 4 THEN H
COLOR= 6: HPLLOT 146,74 TO
146,91
16 220 RETURN
BD 230 HCOLOR= 4: IF I < > 2 * I
NT (I / 2) THEN HCOLOR= 7
46 240 LX = 42 + 21 * X(I):LY =
2 + 18 * Y(I): ON D + 1 G
OTO 250,260,270,280
E8 250 HPLLOT LX + 9,LY + 8 TO LX
+ 9,LY + 17 TO LX + 10,L
Y + 17 TO LX + 10,LY + 8:
RETURN
5F 260 HPLLOT LX,LY + 8 TO LX + 1
0,LY + 8 TO LX + 10,LY +
9 TO LX,LY + 9: RETURN
A8 270 HPLLOT LX + 9,LY TO LX + 9
,LY + 9 TO LX + 10,LY + 9
TO LX + 10,LY: RETURN
11 280 HPLLOT LX + 9,LY + 8 TO LX
+ 20,LY + 8 TO LX + 20,L
Y + 9 TO LX + 9,LY + 9: R
ETURN
A8 290 HCOLOR= 7: IF CP = 2 * IN
T (CP / 2) THEN HCOLOR= 4
7F 300 PX = 43 + X(CP) * 21:PY =
3 + Y(CP) * 18: HPLLOT PX
,PY TO PX + 18,PY TO PX +
18,PY + 15 TO PX,PY + 15
TO PX,PY
BD 310 IF CP = SP THEN GOSUB 330
17 320 RETURN
AD 330 HC = 7:X = X(SP) * 21:Y =
Y(SP) * 18: IF SP = 2 *
INT (SP / 2) THEN HC = 4
CB 340 IF SP = CP THEN HC = 11 -
HC
59 350 HCOLOR= HC: FOR J = 0 TO
1: FOR K = 0 TO 1:PX = 43
+ X + J * 18:PY = 2 + Y
+ K * 16: HPLLOT PX,PY TO
PX,PY + 1: NEXT : NEXT :
RETURN
DB 360 GOSUB 130: CALL 25312,SN(
P) + D,C,X,Y
21 370 RETURN
42 380 R = INT ( RND (1) * 81):
IF BD(R) < > -1 OR R = 4
0 THEN 380
05 390 IF I = L1 THEN L1 = R
35 400 IF I = L2 THEN L2 = R
6E 410 BD(R) = BD(I):BD(I) = - 1
:I = R: GOSUB 110
18 420 RETURN
E5 430 VTAB 7: HTAB 2: NORMAL :
IF MS = 1 THEN INVERSE
A8 440 PRINT "FIRE": NORMAL : IF
MS = 2 THEN INVERSE
05 450 VTAB 9: HTAB 2: PRINT "PA
SS": NORMAL : IF MS = 3 T

```

```

HEN INVERSE
07 460 VTAB 11: HTAB 2: PRINT "Q
UIT": NORMAL : RETURN
A9 470 PX = 44 + 21 * X(I):PY =
4 + 18 * Y(I): HCOLOR= 7:
IF I < > 2 * INT (I / 2)
THEN HCOLOR= 4
F1 480 R = 6: FOR K = 1 TO 18: H
PLOT PX,PY + R TO PX + 15
,PY + R:R = R + 7 - 16 *
(R > 8): NEXT : RETURN
49 490 VTAB 24: HTAB 14: PRINT "
PRESS A KEY":
6B 500 GET A$: HTAB 14: PRINT SP
C(11): RETURN
BA 510 LOMEM: 32768: PRINT CHR$
(21): DIM BD(80),TD(7,3),
X(80),Y(80),P(63),C(63),D
(63)
F6 520 TEXT : HOME : VTAB 4: HTA
B 12: PRINT "LASER CHESS(
TM)": VTAB 7: HTAB 12: PR
INT "COPYRIGHT 1987": HTA
B 7: PRINT "COMPUTE! PUBL
ICATIONS, INC"
E4 530 VTAB 10: HTAB 10: PRINT "
ALL RIGHTS RESERVED"
FD 540 PRINT CHR$ (4);"BLOAD LAS
ER.ML"
91 550 CALL 25315: FOR I = 24400
TO I + 7: POKE I,128: NE
XT
E9 560 POKE 6,80: POKE 7,95: IF
PEEK (191 * 256) = 76 THE
N PRINT CHR$ (4)"pr#a$600
0": GOTO 580
FE 570 POKE 54,0: POKE 55,96: CA
LL 1002
F2 580 FOR I = 0 TO 80:Y(I) = IN
T (I / 9):X(I) = I - Y(I)
* 9: NEXT
E9 590 FOR I = 0 TO 63:C(I) = IN
T (I / 32):P(I) = INT ((I
- 32 * C(I)) / 4):D(I) =
I - 32 * C(I) - 4 * P(I)
: NEXT
2A 600 POKE 28,128: POKE 230,32:
CALL 62454: POKE 49239,0
: POKE 49232,0: POKE 4923
4,0
E6 610 RESTORE : FOR I = 0 TO 8:
READ SN(I): NEXT
BB 620 FOR I = 0 TO 17: READ BD(
I): NEXT
DB 630 FOR I = 18 TO 62:BD(I) =
- 1: NEXT
BA 640 FOR I = 63 TO 80: READ BD
(I): NEXT
CA 650 FOR P = 0 TO 7: FOR D = 0
TO 3: READ TD(P,D): NEXT
: NEXT
71 660 CP = 0:SP = - 1:PP = 1:MC
= 2:L1 = 3:L2 = 77
BD 670 FOR I = 0 TO 80: GOSUB 11
0: NEXT
CB 680 QF = 0:MS = 0:K1 = 0:K2 =
0: GOSUB 430
64 690 IF QF THEN 600
25 700 IF K1 + K2 = 0 THEN 750
78 710 VTAB 23: IF K1 * K2 THEN
HTAB 18: PRINT "DRAW": GO
TO 740
22 720 IF K1 THEN HTAB 14: PRINT
"ORANGE WINS": GOTO 740
12 730 HTAB 15: PRINT "BLUE WINS
"
9F 740 GOSUB 490: GOTO 600
7C 750 IF MC = 2 THEN PP = 1 - P
P:MC = 0:MF = 0: VTAB 20:
HTAB 2: PRINT MID$( " ] ]
[ [ ",PP * 3 + 1,3): IF SP
< > - 1 THEN I = SP:SP =
- 1: GOSUB 110

```

```

5F 770 GOSUB 290
F4 780 GET C$:C = ASC (C$):K = (
C = 11) + 2 * (C = 21) +
3 * (C = 10) + 4 * (C = 8
) + 5 * (C = 13) + 6 * (C
= 44 OR C = 60) + 7 * (C
= 46 OR C = 62) + 8 * (C
= 27): IF C = 0 THEN 780
21 790 OP = CP: ON K GOSUB 800,8
20,840,860,880,1130,1170,
1210:I = OP: GOSUB 110: G
OTO 690
89 800 IF CP > 8 THEN CP = CP -
9
1A 810 RETURN
48 820 IF X(CP) < 8 THEN CP = CP
+ 1
1E 830 RETURN
72 840 IF CP < 72 THEN CP = CP +
9
22 850 RETURN
58 860 IF X(CP) > 0 THEN CP = CP
- 1
26 870 RETURN
26 880 IF SP < > - 1 THEN 920
28 890 IF BD(CP) = - 1 THEN RETU
RN
94 900 I = CP: GOSUB 180: IF PP
= C THEN SC = C:SI = P:SD
= D:SP = CP
18 910 RETURN
58 920 YM = Y(CP) - Y(SP):XM = X
(CP) - X(SP):I = SP: GOSU
B 180:DM = (D < > SD): IF
ABS (XM) + ABS (YM) + DM
+ MC > 2 THEN RETURN
88 930 IF CP = SP THEN 1120
98 940 IF ABS (XM) = 2 AND (BD(S
P + XM / 2) < > - 1 OR SP
+ XM / 2 = 40) THEN RETU
RN
5F 950 IF ABS (YM) = 2 AND (BD(S
P + YM / 2 * 9) < > - 1 O
R SP + YM / 2 * 9 = 40) T
HEN RETURN
88 960 IF YM AND XM AND (BD(SP +
YM * 9) < > - 1 OR SP +
YM * 9 = 40) AND (BD(SP +
XM) < > - 1 OR SP + XM =
40) THEN RETURN
E9 970 IF CP < > 40 THEN 1000
E9 980 IF MF < > 2 THEN MF = 2:I
= SP: GOSUB 380: GOTO 11
20
28 990 RETURN
98 1000 IF BD(CP) = - 1 THEN 109
0
4E 1010 IF (SI < > 0 AND SI < >
6) OR MF = 3 THEN 1070
6A 1020 IF BD(CP) = 0 THEN K1 =
1
57 1030 IF BD(CP) = 32 THEN K2 =
1
CA 1040 IF CP = L0 THEN L0 = - 1
D3 1050 IF CP = L1 THEN L1 = - 1
8F 1060 MF = 3: GOTO 1090
32 1070 IF SI = 2 AND MF < > 1 T
HEN MF = 1:I = CP: GOSUB
380: GOTO 1090
F1 1080 RETURN
7E 1090 BD(CP) = BD(SP):BD(SP) =
- 1
17 1100 IF SP = L1 THEN L1 = CP
20 1110 IF SP = L2 THEN L2 = CP
D1 1120 I = SP:SP = - 1: GOSUB 1
10:MC = MC + ABS (XM) +
ABS (YM) + DM: RETURN
6F 1130 IF SP = - 1 THEN RETURN
3D 1140 I = SP: GOSUB 180:D = D
- 1: IF D < 0 THEN D = S
N(SI + 1) - SN(SI) - 1
29 1150 BD(SP) = SC * 32 + P * 4
+ D: IF I < > CP THEN G

```

```

OSUB 110
EB 1160 RETURN
7F 1170 IF SP = - 1 THEN RETURN
C0 1180 I = SP: GOSUB 180:D = D
+ 1: IF D = SN(SI + 1) -
SN(SI) THEN D = 0
39 1190 BD(SP) = SC * 32 + P * 4
+ D: IF I < > CP THEN G
OSUB 110
D5 1200 RETURN
83 1210 MS = 1
12 1220 GOSUB 430: GET C$: IF C$
= CHR$ (21) OR C$ = CHR
$ (10) THEN MS = MS + 1
- 3 * (MS = 3)
43 1230 IF C$ = CHR$ (8) OR C$ =
CHR$ (11) THEN MS = MS
- 1 + 3 * (MS = 1)
18 1240 IF C$ = CHR$ (27) THEN 1
270
31 1250 IF C$ < > CHR$ (13) THEN
1220
8D 1260 ON MS GOSUB 1280,1590,16
10
43 1270 MS = 0: GOSUB 430: RETUR
N
34 1280 I = (PP = 0) * L1 + (PP
= 1) * L2: IF I = - 1 OR
MF = 4 THEN RETURN
1F 1290 MF = 4:MC = MC + 1:Z = F
RE (0)
AC 1300 LD = BD(I) - 32 * PP - 4
:LB$ = CHR$ (I):LX$ = ""
:EX$ = "":SX$ = ""
88 1310 FOR Z = 0 TO 1 STEP 0: O
N LD + 1 GOTO 1320,1340,
1360,1380
8A 1320 D = 2:J = I - 9: IF I >
8 AND I < > 49 THEN 1400
DD 1330 GOSUB 230: GOTO 1500
6F 1340 D = 3:J = I + 1: IF X(I)
< > 8 AND I < > 39 THEN
1400
E5 1350 GOSUB 230: GOTO 1500
97 1360 D = 0:J = I + 9: IF I <
72 AND I < > 31 THEN 140
0
ED 1370 GOSUB 230: GOTO 1500
5A 1380 D = 1:J = I - 1: IF X(I)
AND I < > 41 THEN 1400
F5 1390 GOSUB 230: GOTO 1500
EE 1400 GOSUB 230:D = LD:I = J:
GOSUB 230
69 1410 I$ = CHR$ (I):LB$ = LB$
+ I$: IF BD(I) = - 1 THE
N NEXT
8C 1420 GOSUB 180:LT = LD - D:LT
= TD(P,LT + 4 * (LT < 0
)): IF LT = - 1 THEN EX$
= EX$ + I$: GOSUB 360:D
= LD: GOSUB 230: GOTO 1
500
E7 1430 IF LT = 4 THEN LX$ = LX$
+ I$ + CHR$ (D - 1 + 4
* (D = 0)):LT = 1
F9 1440 IF P < > 3 OR LT THEN 14
90
D0 1450 IF SX$ = "" THEN 1480
3D 1460 FOR J = 1 TO LEN (SX$):
IF I$ = MID$ (SX$,J,1) T
HEN J = 999
61 1470 NEXT : IF J = 1000 THEN
1500
8A 1480 SX$ = SX$ + I$
7A 1490 LD = D + LT:LD = LD - 4
* (LD > 3): NEXT
13 1500 IF LX$ < > "" THEN I = A
SC (LX$):LD = ASC ( MID$
(LX$,2)):LX$ = MID$ (LX
$,3): NEXT
2D 1510 Z = 1: NEXT : GOSUB 490:
IF EX$ = "" THEN 1570
A5 1520 FOR J = 1 TO LEN (EX$):I

```

```

= ASC ( MID$ (EX$,J)):
GOSUB 470: IF BD(I) = 0
THEN K1 = 1
30 1530 IF BD(I) = 32 THEN K2 =
1
28 1540 IF I = L1 THEN L1 = - 1
AE 1550 IF I = L2 THEN L2 = - 1
F2 1560 BD(I) = - 1: NEXT
88 1570 FOR B = 1 TO LEN (LB$):I
= ASC ( MID$ (LB$,B)):
GOSUB 110: NEXT
FB 1580 RETURN
CE 1590 IF SP < > - 1 THEN I = S
P:SP = - 1: GOSUB 110
0F 1600 MC = 2: RETURN
D0 1610 VTAB 24: HTAB 8: INVERSE
: PRINT "Q";: NORMAL :
PRINT "UIT ";
FE 1620 INVERSE : PRINT "R";: NO
RMAL : PRINT "ESTART OR
";
A4 1630 INVERSE : PRINT "C";: NO
RMAL : PRINT "ANCEL";: G
ET C$: IF C$ = "R" OR C$
= "r" THEN QF = 1: RETU
RN
A2 1640 IF C$ < > "Q" AND C$ < >
"q" THEN HTAB 8: PRINT
SPC( 23);: RETURN
6D 1650 TEXT : HOME : END
F9 1660 DATA 0,1,5,6,10,12,14,18
,22
90 1670 DATA 29,29,16,6,0,8,17,3
0,30
65 1680 DATA 30,26,26,12,20,21,2
6,26,29
2C 1690 DATA 63,56,56,53,52,46,5
6,56,60
AF 1700 DATA 60,60,49,40,32,36,4
8,63,63
FC 1710 DATA -1,-1,-1,-1
81 1720 DATA -1,-1,-1,-1
F3 1730 DATA 0,1,2,3
42 1740 DATA -1,0,4,0
32 1750 DATA 3,2,1,0
E1 1760 DATA 2,1,0,3
95 1770 DATA -1,-1,0,-1
79 1780 DATA -1,-1,1,0

```

Program 5: LASER.ML

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

```

6000: D8 78 85 45 86 46 84 47 ED
6008: A6 07 0A 0A B0 04 10 3E B3
6010: 30 04 10 01 E8 E8 0A 86 81
6018: 1B 18 65 06 85 1A 90 02 31
6020: E6 1B A5 28 85 08 A5 29 13
6028: 29 03 05 E6 85 09 A2 08 EA
6030: A0 00 B1 1A 24 32 30 02 65
6038: 49 7F A4 24 91 08 E6 1A E6
6040: D0 02 E6 1B A5 09 18 69 63
6048: 04 85 09 CA D0 E2 A5 45 DC
6050: A6 46 A4 47 58 4C F0 FD D2
6058: 80 FC E6 E6 FE E6 E6 80 C5
6060: 80 BE E6 E6 BE E6 FE 80 6C
6068: 80 BC E6 86 86 E6 BE 80 AB
6070: 80 BE E6 E6 E6 E6 BE 80 3D
6078: 80 FE 86 86 BE 86 FE 80 01
6080: 80 FE 86 86 BE 86 86 80 18
6088: 80 BC E6 86 F6 E6 BE 80 4F
6090: 80 E6 E6 E6 FE E6 E6 80 78
6098: 80 98 98 98 98 98 80 35
60A0: 80 E0 E0 E0 E0 E6 BC 80 A0
60A8: 80 E6 E6 86 9E E6 E6 80 8A
60B0: 80 86 86 86 86 E6 80 59
60B8: 80 E6 FE E6 E6 E6 E6 80 E2
60C0: 80 BE E6 E6 E6 E6 E6 80 DD
60C8: 80 BC E6 E6 E6 E6 BC 80 11
60D0: 80 BE E6 E6 E6 86 86 80 6A
60D8: 80 BC E6 E6 E6 86 EC 80 C0
60E0: 80 BE E6 E6 BE E6 E6 80 BC

```

60E8: 80 BC E6 8C B0 E6 BE 80 DD
60F0: 80 FE 98 98 98 98 98 80 27
60F8: 80 E6 E6 E6 E6 E6 BE 80 CF
6100: 80 E6 E6 E6 E6 E6 98 80 8C
6108: 80 E6 E6 E6 E6 FE E6 80 91
6110: 80 E6 E6 E6 BC E6 E6 80 E7
6118: 80 E6 E6 E6 BC 98 98 80 1A
6120: 80 FE B0 98 8C 86 FE 80 7F
6128: 95 95 95 95 95 95 95 EA
6130: 00 00 00 00 00 00 00 F2
6138: AA AA AA AA AA AA AA FA
6140: 05 D3 BD DE BF 5F 9D EE C5
6148: EC BF 5F 82 94 BA 52 97 51
6150: 4A 52 94 A5 72 A9 54 01 9D
6158: DC EE 56 BA 9D D2 EA 56 20
6160: EE 77 00 AA 55 28 94 A5 49
6168: 29 4B A5 29 74 A5 00 5F 73
6170: 2F 9C FF 75 EF F7 2F 97 9E
6178: C0 2E 5A D6 B5 AD 6B 5A 75
6180: D6 B5 C0 0A 57 2B 97 AA 16
6188: 95 FA AE 00 75 CE 9D 6B 38
6190: FD AF 5E D7 FB AD D3 AE 74
6198: 00 55 CA FD 54 BD 5C AE EA
61A0: 50 01 7D 2A D7 72 EF 57 DD
61A8: AF 5E AF 5D EB B9 56 BE 58
61B0: 80 AD 5B A7 4A F5 7B 6A 90
61B8: B5 5B F7 EE 70 39 CF DF 03
61C0: AA D5 6D AB D5 EE 9D 2B 87
61C8: 56 00 2E 5C 00 36 D8 6D 40
61D0: B6 DB 60 B9 72 A7 A5 4F 5A
61D8: 4A 9E 95 3D 02 76 B8 4E 9E
61E0: D7 69 DA ED 3B 5D A7 60 23
61E8: 95 3D 2A 7A 54 F4 A9 EB 68
61F0: 97 03 E4 EA 7C 9D 4F 93 18
61F8: A9 F2 75 3E 40 AD 5B A7 F6
6200: 56 EA 9D 6B F9 7D AF 5D 97
6208: 6A 95 7A 76 0E D2 AF 54 8D
6210: BA D7 AF B5 FC EB 75 4E 4A
6218: AD D2 B5 60 F9 2B B7 BE CF
6220: D5 F2 AA 7F 2B A5 56 FA 71
6228: 7E E7 03 9C FD FA A5 D5 3F
6230: D3 F9 55 2B E7 D8 D5 DA EF
6238: 7C 00 35 FF D7 FF FF F7 98
6240: FF DF F7 D7 D7 DF DF F7 6F
6248: D7 FF DF FF FF FF DF D4
6250: FF D7 D7 DF F7 F7 FF FF A3
6258: FF DF FF FF D7 DF FF DF 33
6260: FF D7 D7 D7 FF FF FF F7 8B
6268: D7 DF F7 D7 DF F7 D7 DF FB
6270: FF EB EB FB FF FF AE BA 85
6278: FF AA EA FE BA EA BF AA 68
6280: AF EA FA EE AE FF AA BA AB
6288: FA FF EE BF EA EB FB AB 4F
6290: AF AE AE AF BA BA BB EF D1
6298: AB AB BA BF EA EB EB FA 4B
62A0: FE FF AE BA BB BF B5 BF 6E
62A8: BF B7 B5 B7 BD BF B5 BF 85
62B0: BF B5 B7 BD BF BF BF D1
62B8: BF B5 B7 BD BD BF BF BD C7
62C0: B5 B5 B5 B5 B7 B7 BD BD 85
62C8: BD BD BD BD BF BF BF 9B
62D0: BF BF BF BF BF BF BF 8B
62D8: B7 BF 4C E4 63 4C 14 64 1A
62E0: 4C 5C 64 A9 00 85 E3 18 FF
62E8: A9 38 6D 3A 62 8D B6 63 8C
62F0: 90 07 EE B7 63 EE BD 63 CE
62F8: 18 6D 3A 62 8D BC 63 90 49
6300: 03 EE BD 63 A9 16 85 F9 9C
6308: A9 7C 85 EC A9 65 85 ED 1E
6310: A5 EC 85 EE 48 A5 ED 85 BE
6318: EF 48 20 9A 63 20 31 63 F7
6320: 20 7E 63 68 85 EF 68 85 CB
6328: EE 20 7E 63 C6 F9 D0 E0 15
6330: 60 A9 12 95 EB 0E 7D 63 22
6338: 2E 7C 63 2E 7B 63 B0 1F 6E
6340: A0 FF A2 80 C8 8A 0A B1 E9
6348: EE 2A AA 09 80 91 EC C0 DB
6350: 02 D0 F1 29 BF 91 EC 20 5B
6358: 56 65 C6 EB D0 E2 60 A0 AE
6360: 02 B1 EE 09 40 18 2C B1 6F
6368: EE B0 02 69 80 6A 91 EC 67
6370: 88 10 F4 20 56 65 C6 EB E1
6378: D0 E5 60 2C 3F FC A9 12 43
6380: 85 EB A0 02 B1 EE 49 40 55

6388: 2C B1 EE 49 7F 91 EC 88 EB
6390: 10 F7 20 56 65 C6 EB D0 B5
6398: E9 60 A9 12 85 EB A9 01 F2
63A0: 20 CC 63 90 06 20 CC 63 CD
63A8: 2A D0 F5 AA CA A0 00 BD 88
63B0: 3B 62 91 EC C8 BD 3B 62 C4
63B8: 91 EC C8 BD 3B 62 91 EC EB
63C0: A9 03 A2 EC 20 61 65 C6 58
63C8: EB D0 D3 60 C6 E3 10 11 31
63D0: A2 07 86 E3 AE 40 61 86 79
63D8: 19 EE D5 63 D0 03 EE D6 20
63E0: 63 06 19 60 20 1C 65 8D CD
63E8: 76 65 20 1C 65 8D 77 65 BF
63F0: 20 9E 64 20 D8 64 A2 12 AD
63F8: 8E 79 65 A0 02 B1 FC 91 7E
6400: FE 88 10 F9 A9 03 A2 FC A7
6408: 20 61 65 20 27 65 CE 79 CD
6410: 65 D0 EB 60 20 9E 64 AD D4
6418: 7B 65 C9 04 D0 09 CD 7A 32
6420: 65 D0 04 A9 A0 D0 0A 4D E4
6428: 7A 65 4A A9 80 B0 02 A9 DF
6430: FF 8D 3C 64 A2 12 8E 79 1E
6438: 65 A0 FF A9 00 C8 18 2C F5
6440: 5B 64 F0 01 38 2A 09 80 FA
6448: 91 FE C0 02 D0 EF 29 BF 2A
6450: 91 FE 20 27 65 CE 79 65 D6
6458: D0 DF 60 08 20 1C 65 8D D7
6460: 76 65 20 1C 65 49 01 8D 63
6468: 77 65 20 9E 64 20 D8 64 ED
6470: A2 11 A9 02 2C 77 65 D0 FE
6478: 02 A2 31 8E 87 64 A2 12 1F
6480: BE 79 65 A0 02 B1 FC 11 87
6488: FE 91 FE 88 10 F7 A9 03 54
6490: A2 FC 20 61 65 20 27 65 63
6498: CE 79 65 D0 E6 60 20 1C F5
64A0: 65 8D 7A 65 A0 03 20 FA 71
64A8: 64 18 69 06 48 20 1C 65 97
64B0: 8D 7B 65 A0 B9 C6 64 85 8D
64B8: FE B9 CF 64 85 FF 68 A0 4D
64C0: 00 A2 FE 4C 6B 65 00 00 C7
64C8: 00 80 AB AB 80 50 28 A0
64D0: 31 3A 23 28 31 3A 20 29 83
64D8: A9 7C 85 FC A9 65 85 FD 02
64E0: AD 76 65 A0 D8 20 FA 64 76
64E8: A2 FC 20 6B 65 AD 77 65 33
64F0: A0 36 20 FA 64 A2 FC 4C 3F
64F8: 68 65 A2 08 85 19 84 E3 23
6500: A9 00 85 F9 CA 30 12 0A 35
6508: 26 F9 06 19 90 F6 18 65 AC
6510: E3 90 F1 E6 F9 D0 ED A9 36
6518: 00 A4 F9 60 20 B1 00 20 39
6520: 05 E1 A5 A1 A4 A0 60 18 35
6528: A5 FF 69 04 85 FF 2C 55 0D
6530: 65 F0 01 60 E9 1F 85 FF E6
6538: A5 FE 69 7F 85 FE B0 01 45
6540: 60 E6 FF A5 FF 29 03 F0 EA
6548: 01 60 A5 E6 85 FF A5 FE 45
6550: 69 27 85 FE 60 1C A9 03 04
6558: A2 EE 20 61 65 A9 03 A2 C4
6560: EC 18 75 00 95 00 90 02 26
6568: F6 01 60 18 75 00 95 00 53
6570: 98 75 01 95 01 60 31 20 6A

Save Your Copies of COMPUTE!

Protect your back issues of *COMPUTE!* in durable binders or library cases. Each binder or case is custom-made in flag-blue binding with embossed white lettering. Each holds a year of *COMPUTE!*. Order several and keep your issues of *COMPUTE!* neatly organized for quick reference. (These binders make great gifts, too!)



Binders \$9.95 each;
Cases: \$7.95 each;
3 for \$27.95; 3 for \$21.95;
6 for \$52.95 6 for \$39.95

(Please add \$2.50 per unit for orders outside the U.S.)

Send in your prepaid order with the attached coupon

**COMPUTE!'s
GAZETTE
TOLL FREE
Subscription
Order Line
1-800-247-5470
In IA
1-800-532-1272**

Mail to: Jesse Jones Industries
P.O. Box 5120
Dept. Code COTE
Philadelphia, PA 19141

Please send me _____ COM-
PUTE! cases binders.
Enclosed is my check or money
order for \$ _____ (U.S. funds
only.)

Name _____
Address _____
City _____
State _____ Zip _____

Satisfaction guaranteed or money
refunded.
Please allow 4-6 weeks for delivery.

PARENTS!
TEACHERS!
KIDS!

BE A WINNER IN OCTOBER.

Join Computer Learning Month



WHY A COMPUTER LEARNING MONTH?

Learning with computers is giving America's youth the winning edge. Never before has there been such an exciting event devoted exclusively to bringing parents, teachers and kids together for a month-long celebration of learning with software and computers.

It's called Computer Learning Month, and it's happening this October. Computer Learning Month is so important that a commemorative bill has been introduced into both Houses of Congress declaring October 1987 as Computer Learning Month. Sponsored by major educational software and computer companies and supported by national educational organizations and State Departments of Education, Computer Learning Month will be an event in which everyone will be a winner - an event you won't want to miss.

EVERYONE WILL BE A WINNER...

Contests! Contests! Contests! Thousands of dollars worth of great prizes will be given away in October. Essay and art contests for families and kids. Lesson plan contests for teachers. And if you are a prize winner, your school will be a prize winner, too.

Exciting programs for families. Now parents can learn how to help their children benefit from using a computer, whether they own one or not. We're giving away a free booklet designed just for parents. There will be fairs and special events sponsored by local retailers. And we know

schools will be hosting special parent nights.

A month full of activities for teachers.

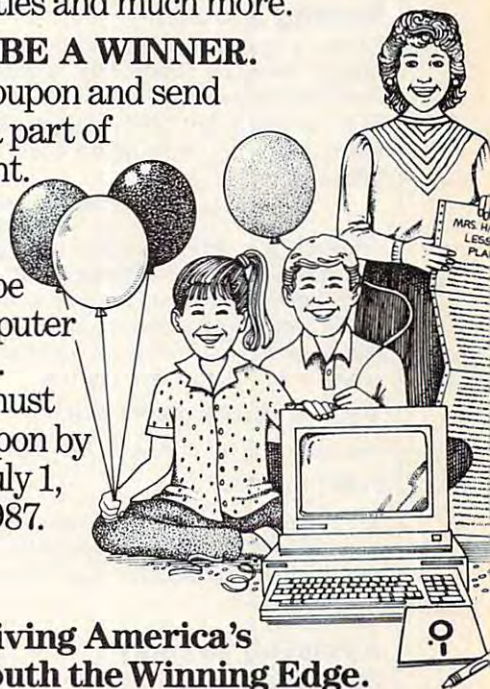
Teachers can join in the celebration easily and learn new ways to help their students benefit from using computers. Every teacher can receive a celebration poster, chock-full of lesson plan ideas, activities and much more.

IT'S EASY TO BE A WINNER.

Just clip this coupon and send it in today to be a part of this national event.

We'll send you everything you need so you can be a winner in Computer Learning Month. Don't delay. We must receive your coupon by

July 1,
1987.



Giving America's Youth the Winning Edge.

YES! I WANT TO BE A WINNER IN OCTOBER!

- Please send me contest rules.
- Please add my name to the Computer Learning Month Mailing List.

I am a teacher parent student

I own a computer yes _____ no
type of computer

NAME _____

SCHOOL _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

PHONE _____

Mail Coupon to: Computer Learning Month
P.O. Box 19763
Washington, D.C. 20036-0763
(202) 452-1600

Computer Learning Month 1987 is Sponsored by: Software Publishers Association, Advanced Ideas, Britannica Software (Publishers of DesignWare, EduWare, Blue Chip Software™), Broderbund Software,™ COMPUTE! Publications, Inc./ABC Publishing, Davidson and Associates, Inc., The Learning Company, Mindscape, Inc., Random House Media, Scholastic, Inc., Spinnaker Software, Springboard Software, Inc., Weekly Reader Family Software.



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.

Rooting Around

I have a question about math on my Commodore 128. I know how to get the computer to give square roots (with the SQR function), but how do I get other roots? For example, how do I get the fifth root of 78?

Neal Hatton

Although your question refers to BASIC 7.0 on a Commodore computer, the answer applies to most computers with BASIC. You can take advantage of the fact that the fifth root of 78 is the same as 78 raised to the 1/5 power. Try this:

```
ROOT = 78^(1/5) : PRINT ROOT
```

You can check your answer like this:

```
PRINT ROOT^5
```

You'll get a number near 78. The number is slightly off because of the limited accuracy of the exponentiation algorithm.

Applesoft Memory Management

I have written an 8K program in Applesoft BASIC and I need to use hi-res graphics page 1, but my program interferes with this page. Can you tell me how to move my program into the open area above page 1?

J. Larry Gaither

For a program stored on disk with the name PROGRAM, include the following as the first line of the program:

```
5 IF PEEK(104)<>64 THEN POKE104
,64:POKE 16384,0:PRINT CHR$(4)
)"LOAD PROGRAM"
```

When you're running a program that uses only the text screen, you have 36,348 bytes of memory available for storing the program and all its variables. But using hi-res screen 1 limits your program and its numeric variables, which are normally

stored before the screen in memory, to 6140 bytes. String variables, which are stored after hi-res screen memory, have 22,016 bytes available. The worst part is that Applesoft BASIC won't tell you when your program and variables fill up their area, as you've no doubt already discovered. If you move your program, all the code and variables will be stored in the larger area above the screen. (Since Applesoft uses 4 bytes to indicate the start and end of program storage, you'll get only 22,012 usable bytes.)

Short programs (less than 6K) which use large array variables might work better if you moved only the numeric variables, leaving the program alone in the lower area. The command to do this is LOMEM: 16384. If the program permits, you might also try switching to hi-res page 2. When using this page, the memory area before the screen is 14,332 bytes long; the area after the screen is 13,824 bytes long. For larger programs, the numeric variables can be moved to the second area with the command LOMEM: 24576.

What You See Is Not...

Why doesn't this program work on my Commodore computer?

```
10 A = SQR(9)
20 IF A = 3 THEN GOTO 10
30 PRINT A
```

When I run this program, it falls through to line 30, printing the value of A, rather than branching back to line 10 as you would expect. Line 30 shows that A is equal to 3, so why doesn't the IF test work?

Christoph Khouri

There are two causes for this discrepancy: the way that the SQR function computes a square root and the way that the computer rounds off some numbers. SQR derives the square root of a number by finding the number's natural logarithm and then dividing that quantity by 2. The result is accurate enough for most practical purposes, but in this case, it yields a number that is very close to 3—but not exactly 3.

When it stores the result of SQR(9) in the variable A, the computer holds the number with a considerable degree of accuracy, including the very small fraction that prevents the result from being exactly

3. The IF test in line 20 compares the number 3—which is exactly 3—with the result of SQR(9), which is a little bit off. Since the two values aren't exactly the same, the IF test fails, and the computer proceeds to line 30.

In that case, you might ask, why does the PRINT statement in line 30 display exactly 3 instead of some fractional value? PRINT is generally accurate, but it rounds off very small fractions. This isn't a defect in the computer, but simply a consequence of the fact that some fractions can't be represented with complete precision in a given numbering system. For instance, in decimal notation, the fraction 1/3 is represented as .33333333... with the 3s carried out an infinite number of places. In this case, the fractional component of the result is so small that PRINT rounds it off rather than attempting to represent it.

Similar anomalies occur in every version of BASIC. Fortunately, few of them cause any real problems unless you are doing serious math which requires a high degree of precision.

Atari Miscellany

I have a few questions about Atari computers. What does the XIO statement do, and what is its syntax? How can I simulate the MOD function? How can I make cartridges for the Atari VCS video-game machine work on my computer?

D. Botha

The XIO statement is a general I/O (Input/Output) statement. Atari BASIC already has many I/O statements such as OPEN, CLOSE, GET, and PUT, but there are some tasks that you cannot do without XIO. For example, the following statement deletes a disk file named TEMP.BAS:

```
XIO 33,#1,0,0,"D:TEMP.BAS"
```

This can be done in program mode or direct mode. You can learn more about XIO in the Atari 400/800 BASIC Reference Manual and similar sources.

MOD is the remainder of an integer (whole number) division. Here is an example of how to translate the expression LOW = LOCATION MOD 256 into Atari BASIC:

```
HI=INT(LOCATION/256)
LOW=LOCATION-HI*256
```

From the publishers of *COMPUTE!*



June 1987 *COMPUTE!* Disk

All the exciting programs from the past three issues of *COMPUTE!* are on one timesaving, error-free, floppy disk that is ready to load on your Atari 400/800, XL, and XE. The June 1987 *COMPUTE!* Disk contains the entertaining and useful Atari programs from the April, May, and June 1987 issues of *COMPUTE!*.

The June 1987 *COMPUTE!* Disk costs \$12.95 plus \$2.00 shipping and handling and is available only from *COMPUTE!* Publications.

For added savings and convenience, you may also subscribe to the *COMPUTE!* Disk. At a cost of only \$39.95 a year (a \$12.00 savings), you'll receive four disks, one every three months. Each disk will contain all the programs for your machine from the previous three issues of *COMPUTE!*. To order a subscription, call toll free 800-247-5470.

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

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

For more information or to order individual issues of the June 1987 *COMPUTE!* Disk, call toll free 1-800-346-6767 (in NY 212-887-8525) or write *COMPUTE!* Disk, P.O. Box 5038, F.D.R. Station, New York, NY 10150.

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,
COMPUTE!'s Apple Applications, and *COMPUTE!*'s Atari ST Disk & Magazine.

Finally, it is impossible to usefully connect VCS cartridges to your Atari computer. Although both devices have a 6502-based microprocessor, their video display hardware is totally incompatible. Rigging up the necessary hardware would probably cost you more than the combined price of a VCS and a 65XE computer.

Reading Amiga Joysticks

I am programming in Microsoft Amiga BASIC, and I want to write a program that uses two joysticks. How can I read a joystick plugged into joystick port 1 (the port where the mouse connector normally goes)?

H. Manson

Amiga BASIC's STICK function works for port 2, but not for port 1. Since the mouse is used to access BASIC's own menus, the designers may have assumed that you would never want to unplug it from port 1. However, it is possible to read a joystick in port 1 by PEEKing a location in memory. Location 14675978 (\$DFF00A) is a 16-bit register that contains the information you need. The position of the joystick is returned in bits 0-1 and 8-9 of this location. Different bits are set to a value of 1 depending on which direction you press the joystick:

```
right: bit 1 = 1
left:  bit 9 = 1
down: (bit 0 XOR bit 1) = 1
up:    (bit 8 XOR bit 9) = 1
```

Location 14675980 (\$DFF00C) contains corresponding information for joystick port 2.

BASIC's BUTTON function also works only for port 2. Here is a program that prints the directions and button status of a joystick in port 1.

```
Stick1:4
jpl% = PEEKW(14675978&)4
bit1% = jpl% AND 24
bit9% = jpl% AND 5124
IF bit1% = 2 THEN PRINT "right ";4
IF bit9% = 512 THEN PRINT "left ";4
IF (jpl% AND 1) XOR bit1%/2 THEN
PRINT "down ";4
IF (jpl% AND 256) XOR bit9%/2 THEN
PRINT "up ";4
IF (PEEK(10952895&) AND 64) = 0 THEN
PRINT "fire";4
PRINT4
GOTO Stick14
```

The last IF statement in the program checks the joystick button in port 1. If for some reason you don't want to use the BUTTON command for port 2, you can test the button with this statement:

```
IF (PEEK(10952895&) AND 128) = 0
THEN PRINT "Port 2 button"4
```

Dynamic Keyboard For The IBM

I have seen a few programs for the Commodore 64 employ a technique you call the dynamic keyboard. The technique adjusts memory locations to make the computer think that someone has typed something on the keyboard. I would like to know how to do this on an IBM PC compatible like my Tandy 1000.

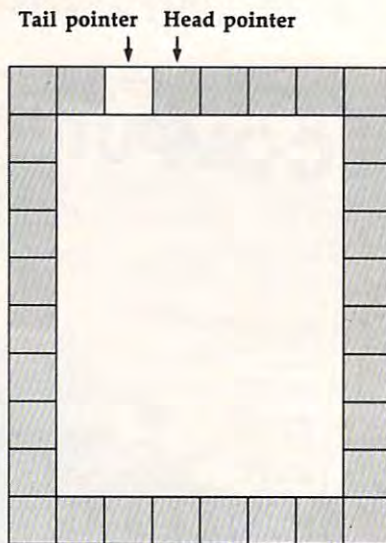
Kevin O'Donovan

The dynamic-keyboard technique can be implemented on most computers which feature a keyboard buffer, including the IBM PC and compatibles. A keyboard buffer is an area of memory where information received from the keyboard is held while awaiting processing. In the IBM PC, PCjr, and true compatibles the buffer normally resides at locations 1054-1085 (&h41E-&h43D). Each time a key is pressed, two bytes are placed into the buffer. The first is the ASCII character code for the key, and the second is the keyboard-scan code for the key. For keys like the function keys which have no corresponding ASCII code, the first byte will be 0, and the second will be the extended keyboard-scan code for the key. Tables of ASCII and extended keyscan codes can be found in your BASIC manual.

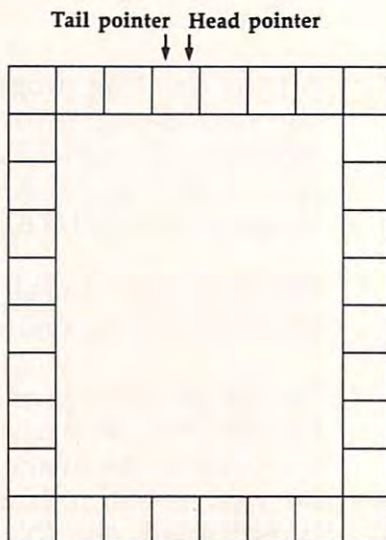
Locations 1050 (&h41A) and 1052 (&h41C) are used as buffer pointers. Location 1050 points to the head of the buffer (the first entry waiting to be read), and location 1052 points to the tail of the buffer (one byte beyond the last entry). Each time a keypress is added to the buffer, the value in location 1052 is incremented by 2. Each time an entry is read from the buffer, the value in 1050 is decremented by 2. The buffer is circular; that is, after a keypress has been recorded in the last two bytes of the buffer, the next keypress will be recorded in the first two bytes. The buffer is considered empty when the value in location 1052 equals the value in 1050, and the buffer is considered full when the value is 1052 is 1 less than the value in 1050. Thus, only 15 of the 16 possible entries in the 32-byte buffer can be used (the maximum number of keypresses the buffer can hold is 15).

To use the dynamic-keyboard technique, POKE the ASCII codes for the desired characters into the buffer. (Remember to use only every other buffer location; it is not necessary to POKE the keycode bytes unless you are trying to enter a key with an extended keyscan code.) If you want the text to be acted upon automatically, remember to include a carriage return (character code 13) as the final character. Then POKE location 1050 with the offset (from location 1024/&h400) to the first character of the entry (use 30/&h1E for the first byte of the

IBM Keyboard Buffer



Buffer full



Buffer empty

buffer) and POKE location 1052 with the offset immediately beyond the last character of your entry. Once you have set up the buffer and pointers, the next time BASIC looks to the keyboard buffer for input, it will find the POKEd characters and process them just as if they had been typed.

One possible use for this technique is to provide default entries for INPUT statements. The following example program illustrates this:

```
100 SCREEN 0:WIDTH 80:CLS:DEF
SEG=0
110 PRINT:PRINT
120 DEFAULT$="Greensboro":GOS
UB 1000
130 INPUT "City: ",CITY$
140 DEFAULT$="NC":GOSUB 1000
150 INPUT "State: ",STATE$
```



```

160 PRINT:PRINT CITY$; ", "; ST
ATE$:PRINT
170 DEFAULT$="Y"+CHR$(29):GOS
UB 1000
180 INPUT "Another entry";OPT
$
190 IF OPT$="Y" OR OPT$="Y" T
HEN 110
200 END
1000 DEFAULT$=LEFT$(DEFAULT$,
15):DBL=2*LEN(DEFAULT$)
1010 FOR II=1 TO DBL STEP 2
1020 POKE &H41D+II,ASC(MID$
(DEFAULT$, (II+1)/2, 1))
1030 NEXT
1040 POKE &H41A, &H1E
1050 POKE &H41C, &H1E+DBL
1060 RETURN

```

The subroutine beginning at line 1000 places whatever is in the variable DEFAULT\$ into the buffer (limited to 15 characters by the LEFT\$ function in 1000), and then sets the buffer pointers appropriately. Line 170 illustrates how cursor-movement characters can be added: The CHR\$(29) is a cursor-left character, which moves the cursor back onto the y after it is printed.

A more common use of the technique is to make programs self-modifying. This scheme relies on BASIC's full-screen editor. You can make a program add lines to itself by printing the desired new program lines at a known location on the screen, and then placing a carriage return (character 13) into the buffer for each line to be added. Here is a very simple address-file program which illustrates this method:

```

10 NEWLINE=300 :ENTRIES=0
20 SCREEN 0:WIDTH 80:CLS
30 PRINT:PRINT"Micro Filer":P
RINT
40 PRINT "Select: [1] Review
addresses":PRINT TAB(9)"[2
] Add addresses":PRINT
50 K$=INKEY$:IF K$="" OR K$<"
1" OR K$>"2" THEN 50
60 IF K$="2" THEN 100
70 RESTORE:FOR I=1 TO ENTRIES
:READ PERSON$,ADDRESS$,CIT
YSTATE$
80 PRINT PERSON$:PRINT ADDRES
S$:PRINT CITYSTATE$:PRINT
90 NEXT:GOTO 30
100 INPUT "Name: ",PERSON
$
110 INPUT "Address: ",ADDRES
S$
120 INPUT "City: ",CITY$
130 INPUT "State: ",STATE$
140 INPUT "Zip code: ",ZIP$
150 CITYSTATE$=CHR$(34)+CITY$
+", "+STATE$+" "+ZIP$+CHR
$(34)
160 CLS:LOCATE 2,1
170 PRINT STR$(NEWLINE)+"data
"+PERSON$+", "+ADDRESS$+"
", "+CITYSTATE$
180 PRINT "10 newline=";NEWLI
NE+10;":entries=";ENTRIES
+1
190 PRINT"run"
200 DEF SEG=0:POKE &H41E,13:P
OKE &H420,13:POKE &H422,1
3

```

```

210 POKE &H41A,&H1E:POKE &H41
C,&H24
220 LOCATE 1,1:END

```

After accepting name and address information, the program creates a DATA statement containing the new entry and prints it on the screen (line 170). It also prints an updated line 10 and a RUN command, then places three return characters into the buffer. Finally, the program relocates the cursor one line above the printed lines and ends. When a BASIC program ends, BASIC prints an Ok prompt, moves the cursor to the next line, and looks for keypresses. In this case, the return characters in the buffer will cause the three printed lines to be entered, and the RUN command in the last of the three lines will restart the newly modified version of the program. The major drawback to using this technique is that all previous variable values are lost when the program is run again. This is why it is necessary to update line 10 to reflect the new values for NEWLINE and ENTRIES.

Refer to Chapter 2 in COMPUTE!'s Mapping the IBM PC and PCjr for more information on the keyboard buffer, including ways to relocate the buffer and increase its size. The example programs should work on most IBM-compatible computers (they were tested on a Tandy 1000EX), but we cannot guarantee that they will work as listed on every variety. It is possible that some machines may place the buffer at locations other than those used by IBM.

Stay Out Of My Zero Page

I am writing a simple graphics routine for the Commodore 64. I have been trying to fill the screen randomly with the ball character, using this BASIC line:

```
10 X=INT(RND(1024)*2023)+1:POK
EX,81:GOTO10
```

The computer puts a few balls on the screen and then locks up. What causes this problem?

Kevin Bentley

You're aiming at the right area (screen memory, locations 1024-2023), but you're using a faulty formula to calculate a random position within that zone. No matter what value you put inside its parentheses, the RND function always returns a fractional value between 0 and 1. The result of RND(1024) is the same as RND(1) or RND with any positive number inside parentheses. Thus, the expression INT(RND(1024)*2023)+1 generates random numbers in the range 1-2023. Some of these POKEs may go in the screen memory area, but many of them will go below the screen, into sensitive memory zones such as the zero page—the lowest 256 bytes of

memory. Nearly all of the locations in the zero page are reserved for the computer's own internal use, and you can easily lock up the machine with careless POKEs in this zone.

Fortunately, no real harm occurs when you crash the computer with a misdirected POKE. Turn the machine off and on to reset it. Here is a program that properly POKEs randomly colored ball characters at random locations on the screen.

```

10 PRINT CHR$(147) CHR$(142)
20 POKE 53280,0:POKE 53281,0
30 JUNK=RND(-TI):BALL=81
40 SCREEN=INT(RND(1)*1000)+102
4
50 COL=INT(RND(1)*15)
60 POKE SCREEN+54272,COL
70 POKE SCREEN, BALL
80 GOTO 40

```

To obtain better randomness, line 30 seeds the random number generator with the expression RND(-TI). The expression in line 40 generates a random number in the range 1024-2023, the correct area for screen POKEs, and line 50 generates a color number in the range 0-15. Line 60 POKEs the randomly selected color value into the color memory location that matches the screen location we chose. Color memory is located at 55296-56295, exactly 54,272 bytes above screen memory. Once the new color has been set, line 70 places the ball on the screen. When POKeing to screen memory, it's usually best to POKE into the matching color memory location first.

Modified Numeric Keypad

Thank you for "Numeric Keypad in 64 Mode" ("Readers' Feedback," October 1986), which allows Commodore 128 owners to use the computer's numeric keypad when in Commodore 64 mode. That program is very useful. However, I would like to change the decimal point on the keypad to a comma, so that when I'm typing DATA statements, I will have the comma in a convenient location.

S. A. Seekins

Load the program, add this line, and save the program under a new name.

```
125 POKE 927,47
```

This POKE makes the period key produce a comma. POKE 927,44 restores the key to normal. ©

Murder Party

Neil Randall

Requirements: Commodore 64, Apple II series, and IBM PC and compatible computers. A printer is required.

As everyone knows, computers provide capable, tireless, and socially unobjectionable opponents for games of all types. After a while, though, it's possible to grow tired of being trounced by computers. And it's also possible to grow tired of playing solitaire games.

Remember those long multiplayer games of the past—before computers—in which socializing was as important as getting the game finished? They had to appeal to the greatest number of people possible. They had to be easy to play, yet challenging enough to satisfy. And they had to have a subject which captivated us.

More often than not, we chose murder.

Murder mysteries. Sherlock Holmes. Hercule Poirot. Perry Mason. Jessica Fletcher. Miss Scarlet in the kitchen with the candlestick. And so on.

Electronic Arts' *Murder Party* allows you quickly and easily to create a party in which you and your guests will try to solve a murder. You need anywhere from five to seven guests. Each of them, in addition to yourself, will play the role of a suspect in the case, with all of you picking up clues from conversations and distributing clues you are given. At the end of the evening, whoever solves the mystery wins the game.

Verdict, Please

The rules are almost nonexistent. The game is divided into four rounds. During each round, every guest must give out some information to the others as discreetly as possible. As host, you try to determine when everyone has done so, and then you move them into the next round. Once the four rounds have ended, the guests fill out a Verdict Sheet, and together you figure out who has won.

Noncomputerized versions of this

type of game exist, but where the computer helps is its ability to keep information hidden from everyone, even the host. In fact, the host does not just run the event; he also plays the game. *Murder Party* scrambles the clues and the murderer's identity, so not even the host knows who did it.

Creating a party is extremely easy. Instructions on the screen take you through the creation step by step, while information in the manual helps you get the party going. First, you choose from one of two scenarios: The Big Kill, or Empire. The disk instructions introduce you to *Murder Party*, then provide information about either scenario. After choosing one, you can then read the invitation that will be sent to the guests, learn more about the suspects to help you assign roles, or, finally, create or edit a party.

To create the party, you fill in the blanks on the screen. You must specify the date, time, and place of the get-together, assign the guests to their roles, and then print out the materials. Casting roles involves choosing which guest will play which role, then entering that guest's name and address. Figuring out who will best suit each part is an extremely interesting exercise, since it means that you should have a reasonable grasp of your guests' personalities.

By Invitation Only

Printing the party materials takes up to two hours with a dot-matrix printer, and it consumes roughly 100 sheets of paper. But it's worth it. *Murder Party* individualizes each packet of information, providing each player with different, sometimes conflicting, information. A few weeks before the party starts, you print out the invitations and envelopes, sending them to the guests without examining them yourself (except your own, of course). Just before the party, you print out the materials for the host (police records, personnel files, coded notes, even a check) and the clues for the guests. The program warns you against the temptation to examine the clues, since doing so will give it all away. Instead, you put them in sealed envelopes and hand them to the guests when the party begins.

The software does all this, while the manual helps you organize the party itself. It provides suggestions and hints for making the guests comfortable (even those who might not enjoy games), and it goes so far as to suggest recipes for food to serve at dinner, music to play in the background, and clothes to capture the ambience. The Big Kill, for instance, is all about a group of ex-Berkeleyites from the sixties getting together just after a friend's death (yes, you've heard this one before), and the manual suggests pulling out your old Woodstock, Canned Heat, or Bob Dylan albums and putting up posters of Joplin and Hendrix. Exceptionally useful, the manual should get you through your first party quite effortlessly. Electronic Arts plans to publish party disks, each with two or more additional scenarios, but the two provided are variable and can be re-used.

For anyone interested in staging such a party, *Murder Party* is exactly right. Not only does it take the drudgery out of producing the needed materials; it also allows you to play the game, not just to organize it and watch it run. Unlike the game master in a standard role-playing game, who spends hours learning rules, drawing maps, and anticipating responses from the players, the host of *Murder Party* simply decides whom to invite, prints out the materials, and plans for an entertaining evening.

Murder Party
Electronic Arts
1820 Gateway Dr.
San Mateo, CA 94404
\$32.95 Commodore 64 version
\$39.95 Apple II series and IBM PC and compatibles versions

Gridiron

Ervin Bobo

Requirements: Amiga or Atari ST. 512K required.

When you've booted *Gridiron* on the Amiga, the title screen presents you with an excellent picture of a ball carrier twisting out of double coverage; the last few bars of "The Star-Spangled Banner" are heard; and the crowd roars. But be ready for a surprise—this is the last you'll see of anything that looks like real football players.

Instead, your view of the field will show you colored dots moving through the play action, resembling nothing so much as an animated chalkboard—but without the arrows. After just a few minutes, you'll realize that this is a pretty good way to play the game.

From opening menus, you choose whether you wish a one- or two-player game, and you set the length of the quarters for 5, 10, or 15 minutes. The shorter times do not allow for a long-range strategy because there will not be as many possessions per team. With the longest quarters, the average game can be played in two hours.

Practice First

Even in the Beginner level, the computer will be a tough opponent. With difficulty being based on factors of speed, strength, and intelligence, you'll want to do a lot of practice before challenging it at any of the three higher levels.

Next you choose whether to use the Standard Playbook or one that you have created using the Play Creation Utility; and whether to use the default team or draft a new team that you have created by altering Player Ability Points. Go through the same steps to select the opposing team, whether controlled by the computer or a friend, and you're ready for the coin toss and the beginning of the game. All selections are made with the mouse. In fact, in playing the game you'll use the keyboard only when entering the names of your teams and the validation code that is a familiar form of copy-protection.

The overhead view will show you about 70 percent of the football field. When action spills across the screen border, the game will momentarily freeze, and then the field will be redrawn to show the missing end. This is a rather nice feature, since scrolling might not allow enough time for you to get the mouse pointer into position—and in this contest you're going to need every break you can get.

In any play, one of your players is controlled by the mouse pointer. He's easy to spot because, a moment before the ball is snapped, he changes color. Hopefully, you'll have your pointer in position, because this player will follow it wherever you drag it—providing no one gets in his way. If you're playing offense, he'll be the quarterback; you'll want to drag him back a bit, then click the left mouse button when you're ready for him to throw. Assuming the pass is completed, the receiver now changes color and will be controlled by the dragging of the pointer, allowing you to weave around the opposing players and make the touchdown—maybe.

It is this single-player control that makes *Gridiron* unique among computerized football games. Others allow some degree of control, but only for the team as a unit, after which you can put your hands in your lap and wait to see how it turns out. *Gridiron*, by allowing this player control, allows you to alter your strategy to fit the moves of the other team.

For setting up a play or a defense, a small box appears in the upper left of the screen and allows you to select a number between 0 and 19; these numbers correspond to the plays in the playbook, or to plays you may have built on your own. While you are doing this, a larger scorecard fills the center of the screen and keeps you from seeing the formation selected by the opposing team. This situation lasts until 20 seconds before the play. If you're on offense, you decide when to snap the ball; otherwise you wait and try to react.

When doing this, you need to remember that all your players will follow the playbook pattern exactly, and that only one man, the one controlled by the mouse, will have the freedom to deviate from the plan.

Play Creation And Player Draft Options

Your own playbook can be created with the Play Creation Utility, in which you can modify either offensive or defensive plays. First, select any of the 20 listed plays from the menu and, if you wish, change the name of the play. Select a player by pointing and clicking; place him on the field; and, from the new edit menu, select the action you wish the player to take. (This explanation is somewhat simplified for space considerations.)

In building players capable of carrying out your schemes—or just better qualified to run the plays in the default playbook—you select the Player Draft option and alter the Player Ability Points for each selected player. Speed and strength of each player is given a



The title screen of *Gridiron* (Amiga version) from Bethesda Softworks.

numerical value that you may raise or lower as you see fit and, during play, the computer takes all this into account: A ball carrier with high strength will not necessarily be brought down or even delayed unless the tackler is of equal strength.

Both the altered players and the altered plays can be saved to a separate data disk for future use. (It occurred to me that you could create a team of wimps who run the wrong way and then force the computer to use that team. So far, it has only occurred to me: It's up to you to figure out the ethics of such a move.)

With the ability to alter players, teams, and the playbook, *Gridiron* presents some options never before seen in such a simulation. The fact that it uses moving dots rather than sprites and blitter objects to represent the players is no handicap to the action and strategy—and may in fact be a blessing for some.

Documentation is very good. And the sound continues throughout the game—not only will you hear the crowd roar; you'll also hear the quarterback making his "hut...hut" calls; hear the grunts as players collide; and, through synthesized speech, hear the referee calling the results of plays.

With all these things going for it, *Gridiron* should become a favorite of die-hard football aficionados as well as offer some appeal for those who just like to play a good game.

Gridiron
Bethesda Softworks
9208 Burning Tree Road
Bethesda, MD 20817
\$59.95 Amiga version
\$49.95 Atari ST version

Destroyer

Scott Thomas

Requirements: Commodore 64, Apple II series with a minimum of 128K, IBM PC and compatibles.

Destroyer, a new game simulation by Epyx, is an interesting blend of strategy and arcade action. Obviously a counter to the numerous submarine simulations that have infested the silicone seas over the last year, *Destroyer* is a simulation of the Fletcher-class US Destroyer from World War II. Besides having some excellent arcade action and strategy elements, the program features some of the best graphics I've seen on a Commodore 64. It puts you in command of a 3000-ton fighting ship with a 325-man crew. Destroyers like the Fletcher-class were the workhorses of the Navy during World War II. Besides providing valuable submarine and aircraft screen for naval and merchant convoys, destroyers ran blockades, rescued downed pilots, bombarded coastal positions in support of invasion forces, and engaged in surface-to-surface combat with other naval vessels. The program provides seven missions of increasing difficulty, each one providing a different combination of tasks that permit you to experience the various activities that destroyers performed during the war.

Destroyer requires a joystick, and loads from a single disk. Due to the numerous graphics screens for the ship stations, the program must access the disk at various points through the game. The program's loading time and subsequent disk-access time, however, is kept to a minimum by the implementation of Epyx's fast *Vorpal* load utility.

Missions And Orders

You begin the program by providing your name and one for the ship in your command. The next step permits you to select from the seven different missions and select from three difficulty levels (easy, intermediate, or advanced). Once your choices are made and your mission selected, you will receive your orders, which describe the mission you must undertake and its objective. Thereafter you enter the navigation station, where plotting the ship's course is accomplished.

With the course plotted, you are then confronted with a screenful of instrumentation known as the Bridge station. From the Bridge, you have a certain amount of control over the other ship stations. Those stations include radar, sonar, surface guns, anti-aircraft guns, torpedoes, and depth charges, as



Destroyer, from Epyx, is a fast-paced blend of action and strategy with excellent graphics.

well as damage control. While you can go to each of these stations by giving a two-letter command, you can receive information from and give certain limited commands to some or all of those stations from the Bridge. Also from the Bridge, as well as from several other stations, you can change the heading and speed of the ship from the plotted course.

It is from the Bridge that you first become aware of the strategy/arcade dichotomy of *Destroyer*. The ship does indeed have a large crew, and all of the stations under your command will, if ordered, operate automatically. On the other hand, you can take manual control of any station—which, with respect to the weapons stations, puts you into an arcade-style action mode. Taking manual control of these stations will be too tempting for some arcade enthusiasts, as the graphics and action are very good. However, even arcade enthusiasts will quickly learn how fast a vessel can be destroyed and the mission lost if they spend too much time manually shooting down aircraft or torpedoing enemy ships.

The reverse situation also is true. You cannot command your ship from the Bridge and succeed in your mission. While your crew can perform with minimum levels of adequacy at each station, an attack on your vessel or a command by you requiring the capabilities of a particular station will most often require your direct attention for a period of time. Accordingly, *Destroyer* demands a precise blend of the commander's use of the strategy and action aspects to succeed on the mission.

One of the most interesting and challenging stations is damage control. By typing the letters DA, you are transferred to the damage-control station, which gives you a cross-section view of your ship and a list of the areas of the ship that can incur damage. Also listed are four damage-control parties, identified as Alpha, Baker, Charlie, and Delta teams. The teams are listed in order of

their speed of repair, with Alpha team being the fastest. When damage is incurred, the teams are assigned automatically in descending order to fix the repairs. However, you can manually reassign the parties to any damaged stations. The reassignment capability will be critical to your success—particularly in the more difficult missions, where some damage to your ship is almost certain—since the automatic assignment routine may assign the fastest teams to the least important stations for repairs. Depending upon your mission and the circumstances of the moment, the repair of damage to the steering system or sonar may be much more critical than damage to the surface guns or the torpedo stations.

The instruction manual that accompanies the program is brief, but complete and informative. It includes a breakdown of each station, as well as some interesting and useful information about the Fletcher-class Destroyer.

In summary, *Destroyer* is one of those hybrid strategy/arcade simulation games that excels by performing well in both areas and by requiring skills in both areas for player success. The program maintains Epyx's reputation for superb graphics, and is fast-paced, with plenty of action. If you fancy yourself to have "the right stuff" to be captain of a fighting ship, *Destroyer* can certainly provide you with a taste of the rewards and rigors of command.

Destroyer

Epyx

P.O. Box 8020

600 Galveston Rd.

Redwood City, CA 94063

\$39.95 all versions

All programs
listed in this
magazine are
available on the
GAZETTE Disk.
Details
elsewhere
in this issue.

SILICON EXPRESS DISKS
SS/IDD 6.95
DS/IDD 7.95

VERBATIM OR MAXELL
SS/IDD 13.95
DS/IDD 16.95

ELEPHANT
SS/IDD 7.95
DS/IDD 8.95

DATA CASE
7.95

INSIDE OHIO
1-800-225-0755
YOUR TOLL FREE ACCESS OUTSIDE OHIO
1-800-228-0755
CALL...FOR THE BEST PRICES...
BEST DELIVERY...
We have hundreds of other programs in stock at Fantastic savings.

SILICON EXPRESS

LASER 128—APPLE COMPATIBLE COMPUTER \$389.00

APPLE SOFTWARE		APPLE SOFTWARE	
Algebra	27.89	Adv. Construction	33.89
American Challenge	24.89	Airheart	22.89
ASC II Express	74.89	Algebra	27.89
Balance of Power	33.89	Alpha Plot	20.89
Bank Street Writer+	58.89	ASC II Express	74.89
Breakers	28.89	Auto Works	25.89
Certificate Maker	41.89	Bag of Tricks 2	31.89
Library Vol. 1	19.89	Bank Street Filer	42.89
dBase III	419.89	Bank Street Mailer	42.89
Dollars & Sense	104.89	Bank Street Writer	42.89
Easy	88.89	Bank Street Speller	42.89
Flight Sim	28.49	Bard's Tale	39.89
Fontasy	88.89	Beagle Bag	16.89
Fntpk 1 thru 13	ea. 13.89	Beagle Basic	18.89
Frameworks II	419.89	Beagle Compiler	41.89
Fully Powered PC	24.89	Beagle Graphics	30.89
Gamecard III	39.89	Beneath Apple DOS	12.89
Gato	21.89	Big U	18.49
Hacker I or II	24.89	Bookends	66.89
Helicat Ace	24.89	Bop & Wrestle	18.89
Jet	31.49	Captain Goodnight	22.89
King's Quest I or II	31.89	Carmen Sandiego	25.89
Load Runner	22.89	Certificate Maker	29.89
Locksmith	56.89	Library Vol. 1	19.89
Lotus	319.89	Chart'n Graph Tlbr	23.89
Mean 18	34.89	Classmates	27.89
Mngng Yr Money	114.89	Database Toolbox	23.89
Micro Cookbook	27.89	Dazzle Draw	37.89
Microsoft Word	279.00	D-Code	20.89
Multipan	110.89	DiskQuick	15.89
Newsroom Pro	74.89	Dollars & Sense	71.89
Norton Utilities	58.89	DOS Boss	12.89
Option Board	79.89	Double-Take	18.49
PC Tools	18.89	Eidolon	24.89
PFS: Access	59.89	Essential Data Dup	20.89
PFS: Files Profes.	148.89	Extra K	20.89
PFS: First Choice	104.89	Fantavision	31.89
PFS: Plan Profes.	148.89	Fat Cat	18.49
PFS: Write Profes.	148.89	Flex Type	15.49
Print Shop	34.89	Flight Sim II	34.89
Print Shop Companion	32.89	Fontrix 1.5	58.89
Print Shop Gr. #1	21.89	Fntpk 1 thru 15	ea. 13.89
Reflex Database	86.89	Font Works	33.89
Report Card	32.89	Forecast	37.89
Sidekick	50.89	Frame-Up	15.49
Sidekicks Ver. #3	38.89	Gato	21.89
Silent Service	22.89	G I Joe	24.89
Speedkey	104.89	Graphics Department	28.89
SuperCalc 4	279.89	Graphic Magician	25.89
Superkey	40.89	Graphworks	44.89
Symphony	494.89	GPLe	25.89
Think Tank	99.89	Handler Package	58.89
Toy Shop	46.89	Impossible Mission	24.89
Traveling Sidekick	40.89	I.O. Silver	15.89
Turbo Database Tlbr	40.89	Jeeves	28.89
Turbo Lighting	58.89	Karateka	22.89
Turbo Pascal 3.0	59.89	Knight of Diamonds	21.89
Turbo Prolog	58.89	Kid Writer	24.89
Turbo Tutor	23.89	King's Quest I, II or III	31.89
Ultima III	40.89	Koronis Rift	24.89
Visable 8088	47.89	Legacy of Lylgamy	24.89
Volkswriter Dix #3	141.00	List Handler	36.89
Web. Spell Checker	39.89	Locksmith	38.89
Wizardry	36.89	Macroworks	18.89
Word Perfect	289.00	Mngng Yr Money	98.89
Wordstar 2000+	284.89	Megaworks	CALL
		Merlin	41.89
		Merlin Pro	64.89
		Merlin Combo	68.89
		Micro Cookbook	27.89
		Millionaire	31.89
		Minipix 1.2 or 3	15.49
		Mix Choice	24.89

SILICON EXPRESS
5955 E. Main St. Columbus, Ohio 43213
1-614-868-6868

COMMODORE		HARDWARE	
Acrojet	22.89	CompuServe Starter	24.89
Air Rescue I	24.89	Datasec	7.89
Bard's Tale	30.89	Disk Notcher	4.89
Ballblazer	24.89	Apple Flip-n-File	10.89
Back to Basics	134.89	Lemon	27.89
Bank Street Storybook	27.89	Time	48.89
Bank Street Speller	34.89	APPLE HARDWARE	
Bank Street Writer	34.89	80 Col. 64K Card (II E)	49.89
Basic Toolkit	29.89	Apple II	184.89
Blazing Paddles	24.89	Disk Drive Cont	47.00
Borrowed Time	20.89	Disk Drive HI Tech	119.89
Carmen Sandiego	22.89	Gibson Light Pen	148.00
Certificate Maker	29.89	Grappler Buffered	118.89
Color Me	20.89	Grappler Pro	74.89
Cntdwn to Shtdwn	20.89	Grappler (serial)	74.89
Fast Tracks	20.89	Grappler C	74.89
Gato	20.89	Hotlink	44.89
Gertrude's Secrets	20.89	Kraft Joystick IIE IIC	23.89
G I Joe	24.89	Koala Pad +	78.89
Graphics Magician	27.89	Laser 128 Computer	369.00
Grt Amer Road Race	19.89	Mach II Joystick	28.89
Hacker	20.89	Mach III Joystick	34.89
Jet	27.89	Mach IV Joystick	59.89
Jet Combat Simulator	20.89	Micromodem IIE	139.89
Karateka	22.89	Mockingboard A or B	64.89
Kennedy Approach	24.89	Mockingboard C	114.89
Kung Fu	20.89	Multiram CX 512K	174.00
Little Comp. People	24.89	Multiram IIE 80/64K	119.00
Mail Order Monsters	24.89	Mouse House	7.49
Merlin	34.89	Mouse Mat	6.49
Mig Alley Ace	24.89	Paddlesticks	26.50
Multipan 64/128	43.89	Parallel Printer Card	44.89
Nato Commander	22.89	Prometheus 1200A	254.89
Printer Clip	41.89	Print-it	122.89
Print Shop	29.89	Ramfactor 512K	219.89
Print Shop Comp.	21.89	Ramfactor 1 MEG	289.89
Pr. Shop Gr. 1,2, or 3	15.89	Ramworks 64K	139.89
Rescue on Fractalus	24.89	Ramworks 512K	199.89
Scrabble	27.89	Ramworks 1 MEG	269.89
Sidekicks	20.89	Reader Rabbit	24.89
Silent Service	22.89	Sargon III	31.89
Spitfire Ace	20.89	Sat (Harcort Brace)	27.89
Spreadsheet	34.89	Sorcerer	27.89
Sticky Bear ABC	20.89	Speed Reader II	38.89
Numbers	20.89	Spellbreaker	30.89
Opposites	20.89	Spell II	27.89
Shapes	20.89	Starcross	30.89
Summer Games I or II	24.89	Star Trek I or II	27.89
Temple of Apshai Trl.	24.89	Suspect	27.89
Toy Shop	39.89	Tass Times	24.89
Transformers	24.89	Ultima III	40.89
World Games	24.89	Temple of Apshai Trl	24.89
World's Grtst Bsbll	24.89	Trinity	20.89
World's Grtst Ftbll	24.89	Typing Tutor III	31.89
		Will Writer	28.89
		Winter Games	24.89
		Wishbringer	24.89
		Witness	24.89
		Word Attack	27.89
		World's Grtst Bsbll	24.89
		Zork I	24.89
		Zork II or III	27.89
		Zork Trilogy	46.89

SILICON EXPRESS
5955 E. Main St. Columbus, Ohio 43213

Name _____
Address _____
City _____ State _____ Zip _____

Charge # _____ Exp. Date _____

QTY.	DESCRIPTION	PRICE

Computer Type _____ SHIPPING _____
Phone No. _____ TOTAL _____

Add \$3.00 min. U.S. shipping, C.O.D. \$5.00 extra. Hawaii and Alaska \$4.95 min. Orders outside U.S. are not insured. Canada & Mexico 10% min. \$10.00. All other countries 25% min. \$30.00. MasterCard, Visa and school purchase orders accepted. Personal checks allow 3 weeks. 5.5% sales tax for Ohio residents. Defective replaced within 20 days. 20% restocking fee if not replaced with same item. Compatibility not guaranteed. Prices subject to change without notice.

Defender Of The Crown For Amiga

Neil Randall

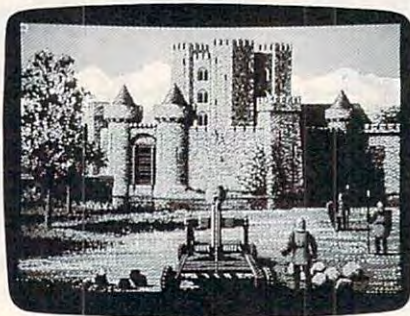
Requirements: Amiga with 512K minimum of memory.

Defender of the Crown is the Amiga introduction to Mindscape's new Cinemaware series. Labeled "interactive movies," the Cinemaware products attempt to capture the heroism of Hollywood in the form of a graphics adventure. *Defender of the Crown* places you in the role of an English noble in the swashbuckling world of Robin Hood, a world in which you lead armies, joust in tournaments, and rescue ladies.

By far the strongest feature of the Cinemaware products is the exceptional graphics. From its opening credits screen to the spectacle of the tournament, *Defender* makes full use of the Amiga's enormous graphics potential. The pictures almost leap off the monitor with clarity, color, and sheer beauty, and the animation is frequently good enough to achieve the game's goal of having the viewer feel part of a movie. The 3-D map of Britain, the pictures of the castles, the splendid portrayal of the tournament—all represent the finest artwork produced so far in a microcomputer game.

At various points in the story, for example, you set out to rescue a Saxon lady from the dastardly Normans. The game places your character in the courtyard of a castle, in which each brick on the wall and each star in the sky is clearly delineated. Your character, along with two allies, advances toward three defending swordsmen. If you win the sword fight, you move inside the castle, thrusting, slashing, and parrying your way up a flight of stone steps as the torches flicker on the walls behind you. Once past this guard, you enter the castle's interior, where the frightened damsel awaits you. The sequence places you, finally, in front of a fireplace, and you watch two silhouettes embrace and kiss. True to old Hollywood, the scene tastefully fades.

Unfortunately, *Defender's* gameplay is less impressive. In one turn you can recruit your home army, build the campaign army, ride in search of conquest, besiege an enemy castle, or launch a raid against an enemy's possessions. You are constantly endangered by the armies of the other nobles, and as the game moves to a contest between you and one other noble, the opposing army can grow very large indeed. But where a good war game would use these medieval essentials as the basis for a thoughtful simulation,



Mindscape's *Defender of the Crown* makes full use of the Amiga's graphics potential.

Defender oversimplifies each step. Power politics, in the form of alliances and treachery, are nowhere to be seen, and battles offer nothing of the strategies and tactics of the period. Even the jousting and sword fighting, on which the game's action relies, are not particularly enjoyable. *Defender of the Crown* rests on its graphics, and this fact is both its strength and its weakness.

These criticisms are perhaps unfair, since *Defender* is not an attempt at historical simulation. But after the initial dazzle of the graphic excellence subsides a bit (it never leaves completely), the game isn't nearly as enjoyable as it could be. It is fun, and its historical flavor is undeniable, but it leaves me with a strong hope that future Cinemaware productions will complement the enormous graphic success with a more solid and replayable game. *Defender of the Crown* is a must for any Amiga owner, because with it you begin to understand your computer's capabilities. But it could have been even better. When superb graphics and superb game finally merge, as they surely will in a future Cinemaware product, computer gaming will never be the same again. For now, though, buy *Defender*. As Hollywood fantasy, it's as good as you can get.

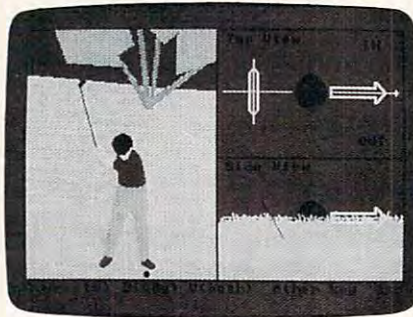
Defender of the Crown
Mindscape
3444 Dundee Rd.
Northbrook, IL 60062
\$49.95

Championship Golf

Neil Randall

Requirements: Amiga (reviewed here), and IBM PC and compatible computers.

Among the sports games developed early in the Amiga's life cycle, golfing simulations seem to dominate. *Leader Board* and *Mean 18* were first off the tee, each with its own touches of excellence and each gaining its own following. Now into the race comes *Gamestar* (a subsidiary of Activision), whose *Star League Baseball* and *On-Court Tennis* showed Commodore 64 and eight-bit Atari owners what their machines were capable of. *Championship Golf* for the Amiga goes a long way toward demonstrating the kind of simulations we can begin to expect on the new, higher-powered personal computers.



Graphics, animation, and excellent playability recommend *Gamestar's* *Championship Golf* (PC version shown here).

Remarkable Graphics

Championship Golf uses a split screen to simulate the intricacies of the famous Pebble Beach course. On the left is an overhead shot of the hole you are playing, while the right side shows the same hole from the golfer's ground-level point of view. Both views are variable, though; the function keys let you zoom in on the overhead screen and step back to a bird's-eye view on the other. The graphics detail on both screens is remarkable, but especially interesting are the fractal techniques used in the ground-level view. Hills and slopes are all visible, and the complexity of the course becomes obvious. In fact, you can position the golfer anywhere you like on the hole, and the graphics will be redrawn to show your new perspective. In this way, you can walk the course.

Once you've aimed your shot, the left screen changes to allow you to choose the club you will use and the

type of shot you will take (full swing, chip, or putt). For all shots, the computer acts as your caddy, suggesting club and shot type. After this, you set your stance and the alignment of the clubface, working with the wind to hook or slice your shot. Finally, the golfer appears, ready to swing away (under your control). The graphics are superb here, with the golfer's animation clear and accurate right down to wrist movement, and once the shot is taken, you watch the ball in the ground-level screen. All in all, the graphics are fully worthy of the Amiga's capabilities.

In addition to addressing stance and clubface alignment, you have the option of adding arm speed, wrist action, and body action as you take the shot. As in real golf, these activities must be learned and practiced, and also as in real golf, success makes you feel extremely good. After some initial ineptness in which you watch shot after shot dribble off the tee, you will suddenly see the golfer get under the ball perfectly, and you will watch the ball disappear above the height of the fairway than you've managed so far. With the right amount of practice, you can make these shots fairly regularly.

Game Options

To help you perfect your swing, *Championship Golf* includes a driving range. Here you can practice teeing off, hitting from the rough or off the fairway, chipping out of two types of bunkers, or putting on a practice green. The driving range is a must for those wanting to become expert players, even those experienced in real golf. While *Championship Golf* seems to be realistic, it is inevitably different from actually swinging a golf club.

The game offers several other options. You can hit from three different tee placements, you can carry clubs not normally used (2-wood and 1-iron, for example), and you can save the current game to disk. Since the full Pebble Beach course takes a fairly long time to play through, saving is extremely convenient. For those who wish to see how the game works, pressing RETURN and F10 through the shot sequence will provide a perfectly respectable, but hardly awe-inspiring, round.

Championship Golf is subtitled *Volume One: Pebble Beach*. The implication, clearly, is that Gamestar intends to issue other volumes, each with its own course. If each course is as detailed as Pebble Beach, I can easily imagine a golfing fan happily buying every volume.

The system's only drawbacks are the amount of disk access, the time taken to redraw the ground-level views, and the strangely undetailed putting game. The gamemakers could help disk access by allowing those with extra memory to load parts of the program into RAM; they could also improve the putting. Whether or not Gamestar can speed up the redrawing is relatively unimportant; the ground-level and bird's-eye views are spectacular, and well worth the wait. *Championship Golf* is indeed a very good game.

Championship Golf
Gamestar
 P.O. Box 7287
 Mountain View, CA 94039
 \$39.95 Amiga and MS-DOS versions

We Do Windows...Quickly!





C-MORE

An Operating System for the Commodore 64.

- USE YOUR COMMODORE LIKE A PC!
- Let C-More teach you all about business computing! Control multiple applications in windows. Word Processing, Spreadsheet, Database, Scientific Calculator, Disk Utility and Comparative-Buyer programs are all included!
- Comprehensive 450-page manual provides tutorials, step-by-step instructions, examples and illustrations, for any level user!
- WRITE COMMODORE BASIC PROGRAMS FOR C-MORE!!

1-800-628-2828 ext. 790

 Visa and Mastercard Accepted 

or mail \$49.95 plus \$4.50 shipping & handling
 (\$6.00 shipping in Canada)

(Georgia residents add \$2.00 sales tax)

C-More Products, P.O. Box 81548 Chamblee, GA 30366

Please allow 4-6 weeks for delivery.

"Commodore" is a trademark of Commodore Electronics, Ltd.

THE AMAZING VOICE MASTER

ENTER
 THE FINAL
 FRONTIER
 OF
 MAN-TO-MACHINE
 COMMUNICATIONS



There is nothing else like it. Voice Master gives both speech output and voice recognition with this single hardware product! Your voice controls programs, or home appliances, robots, and more with spoken commands. Verbal response back gives status, verifies, or requests your reply! Speech output and recognition patterns are recorded in with your voice. Or use the voice of your friend, boss, teacher, mother, even the family pet! Programming is simple with new commands added to BASIC. A music bonus program lets you write and compose musical scores simply by humming the tune. Unlimited applications for fun, education, and commercial use. Design your own programs for profit. Speech and recognition quality unsurpassed by even the most sophisticated machines. Only Covox provides this high-tech marvel at a price less than most common peripherals.

The Covox Voice Master comes complete with all hardware and software for only \$89.95. (Add \$4 shipping and handling for USA, \$6 Canada, \$10 overseas.) Available for Commodore 64/128, Apple II, II+, IIc, IIe, Atari 800, 800XL, 130 XE. Specify when ordering. Visa, MasterCard phone orders accepted.



Call or write for FREE Voice Master Infopak and special combination package offers.

COVOX INC., DEPT. C!

675-D Conger Street • Eugene, Oregon 97402 • U.S.A.
 Area Code (503) 342-1271 • Telex 706017 (Av Alarm UD)

Super Hi-Res Graphics And Sound On The Apple IIGS

William B. Sanders

The IIGS, the newest computer in the Apple II line, is aptly named. Its superb graphics and sound—hence, GS—make this computer the most exciting machine in the Apple II line. Commercial software developers are moving quickly to take advantage of these features. For amateur programmers, the GS also opens up new areas to explore. Here's an overview of the IIGS's amazing super high-resolution color graphics and powerful sound capabilities, excerpted from William B. Sanders' recently published book, The Elementary Apple IIGS, from COMPUTE! Books.

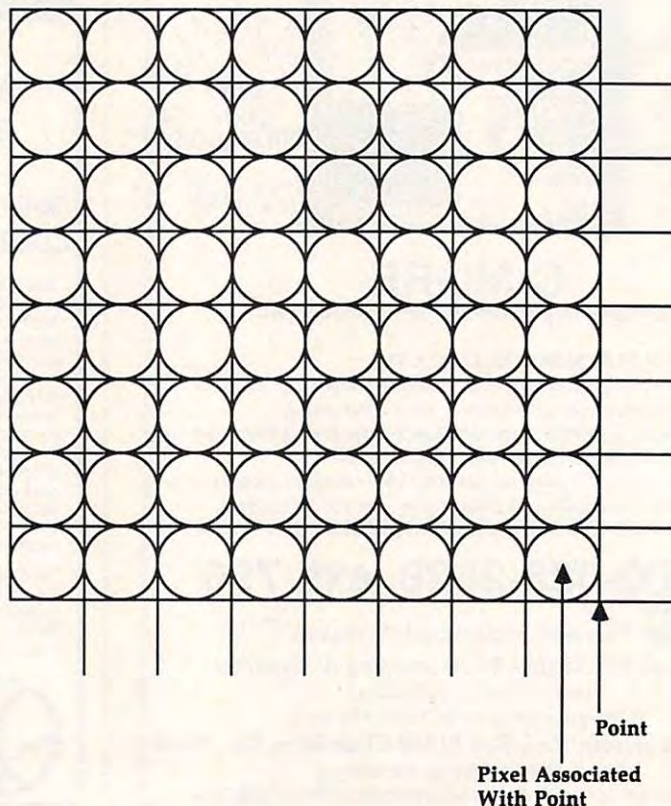
Your IIGS has made a quantum leap over the other Apple IIs when it comes to graphics and sound. The reason is that it can place pixels on a 640×200 -dot matrix for four-color programs to gain the highest resolution on an Apple II yet. And it has an Ensoniq sound chip for creating unsurpassed digitized sound. There are some excellent programs available for accessing these new features, but you cannot easily work with them directly from Applesoft BASIC. Applesoft BASIC remains the same as it has always been, so the thousands of programs written in it can also run on the IIGS.

Apple has made it simple for advanced programmers to use the super high-resolution graphics and

sound. However, because of the advanced skills required for accessing the "toolboxes," we'll introduce just the concepts here so that you can get an idea of whether you're interested in pursuing the advanced programming levels required to ac-

cess the IIGS's super high resolution and sound. Remember, there are application programs available for you to do all of this without any programming skills at all. However, you might find doing the programming yourself to be a real adventure.

Figure 1: Pixel Matrix



The QuickDraw II Toolbox: Super Hi-Res Graphics

The collective set of the subroutines built into the IIGS ROM have been named QuickDraw II. By making calls to these routines, you can create lines, rectangles, polygons, arcs, and other figures. With the toolbox routines in QuickDraw II, much of the work has been done for you.

The basic building block of the routines is the pixel matrix made up of points and associated pixels. Each point referenced through the toolbox routines has an associated pixel above and to the left of the point. A pattern is composed of an 8 × 8 matrix of points and pixels (Figure 1).

Depending on the values associated with a given pixel matrix, different patterns, shapes, and figures can be created. The boundaries of the drawings created in QuickDraw II are (-16384,-16384) and (16383,16383) on a Cartesian matrix with memory space of 32768 × 32768 pixels. Various calls made in assembly language, C, or another language that can easily access the QuickDraw II tools are used. The calls are given various names associated with addresses in memory. Let's look at some of these calls to get an idea of what is available in the graphics toolbox inside your IIGS.

QuickDraw II Calls

The first set of functions in the QuickDraw II toolbox is intended for housekeeping purposes. These functions set up the various registers and pointers to allow access to the graphics tools. They include QDBootInit, which initializes the QuickDraw II tools when the system is booted; QDStartup, which initializes QuickDraw II and sets the standard port and clears the screen; and QDShutdown, which turns QuickDraw II off and frees the buffers. QDVersion and QDStatus, respectively, provide information on the version of QuickDraw II and specify whether or not it is active.

A second set of QuickDraw II routines allows you to control the characteristics of the screen display. Each of the 200 horizontal scan lines that make up the display has its own scan-line control byte

(SCB), so the characteristics of each line can be specified independently. The GetStandardSCB routine returns information about the SCB. The first four bits (0-3) are used for color table 0; bit 4 is reserved; bit 5 controls the fill option; bit 6 specifies whether an interrupt can occur; and bit 7 controls the color mode (320 pixels or 640 pixels). The call SetMasterSCB sets the low byte of the master SCB, and GetMasterSCB returns the same information. SetSCB, GetSCB, and SetAllSCBs are further scan-line control-byte calls. For setting the color table, either in the 320 or 640 mode, the InitColorTable call is used. Table 1 lists the values (in hexadecimal) for the two modes.

Calls to SetColorTable, Get-

ColorTable, SetColorEntry, and GetColorEntry all access the routines to set and get information about the colors.

The other global calls deal with the fonts, clearing the screen, and turning the super hi-res graphics mode on and off. The calls include SetSysFont, GetSysFont, ClearScreen, GrafOn, and GrafOff. The functions of the calls are fairly self-explanatory, making it easier to use the graphics than to use more obtuse codes.

Port Of Calls

Now you have some idea of a few of the functions and how the name of the call is connected to the functions. For the rest of the QuickDraw II functions, refer to Table 2, just to

Table 1: Hexadecimal Values For Setting Color Table

320 Mode			640 Mode		
Pixel	Color	Code	Pixel	Color	Code
\$0	Black	0 0 0	\$0	Black	0 0 0
\$1	Dark gray	7 7 7	\$1	Red	F 0 0
\$2	Brown	8 4 1	\$2	Green	0 F 0
\$3	Purple	7 2 C	\$3	White	F F F
\$4	Blue	0 0 F	\$4	Black	0 0 0
\$5	Dark green	0 8 0	\$5	Blue	0 0 F
\$6	Orange	F 7 0	\$6	Yellow	F F 0
\$7	Red	D 0 0	\$7	White	F F F
\$8	Flesh	F A 9	\$8	Black	0 0 0
\$9	Yellow	F F 0	\$9	Red	F 0 0
\$A	Green	0 E 0	\$A	Green	0 F 0
\$B	Light blue	4 D F	\$B	White	F F F
\$C	Lilac	D A F	\$C	Black	0 0 0
\$D	Periwinkle blue	7 8 F	\$D	Blue	0 0 F
\$E	Light gray	C C C	\$E	Yellow	F F 0
			\$F	White	F F F

Table 2: GrafPort Calls

OpenPort	InitPort	ClosePort	SetPort	GetPort
SetPortLoc	GetPortLoc	SetPortRect	GetPortRect	SetPortSize
MovePortTo	SetOrigin	SetClip	GetClip	ClipRect
HidePen	ShowPen	GetPen	SetPenState	GetPenState
SetPenSize	GetPenSize	SetPenMode	GetPenMode	SetPenPat
GetPenPat	SetSolidPenPat	SetPenMask	GetPenMask	SetBackPat
GetRgnPat	SetPolyBackPat	SolidPattern	PenNormal	MoveTo
Move	SetFont	GetFont	SetFontID	GetFontID
GetFontInfo	GetFGSize	GetFontGlobals	SetFontFlags	GetFontFlags
SetTextFace	GetTextFace	SetTextMode	GetTextMode	SetSpaceExtra
GetSpaceExtra	SetCharExtra	GetSpaceExtra	SetForeColor	GetForeColor
SetBackColor	GetBackColor	SetBufDims	ForceBufDims	SaveBufDims
RestoreBufDims	SetClipHandle	GetClipHandle	SetVisRgn	GetVisRgn
SetVisHandle	GetVisHandle	SetPicSave	GetPicSave	SetRgnSave
GetRgnSave	SetPolySave	GetPolySave	SetGrafProcs	GetGrafProcs
SetUserField	GetUserField	SetSysField	GetSysField	

see the extent of the toolbox. Further description here would do little good, since you may not yet have the programming skills necessary to use the calls.

Drawing Calls

The table below will give you a better idea of what kinds of drawing shapes are supported by the QuickDraw II toolbox. This set of calls is what most programmers will use often when they're creating graphics directly or when they're writing a graphics drawing program.

By looking over these calls from the QuickDraw II toolbox, you can become acquainted with what is available and get a clue as to the graphics power of your Apple IIGS computer.

The Sound Of The IIGS

The Apple IIGS houses a 5503 Ensoniq Digital Oscillator Chip (DOC). The DOC has 32 digital oscillators that give you everything from beeps and buzzes to a talking computer and symphonic orchestra. However, as with the super high-resolution graphics, you must

use the DOC toolbox to take full advantage of this feature.

To get started, let's look quickly at the registers used to control the sounds in DOC.

Frequency control (low and high). Two registers control frequency; joined together they form a 16-bit value used for the 24-bit accumulator. The value of this register pair is added to the current value stored in the 24-bit accumulator.

Address: \$00-\$1F (low)
\$20-\$3F (high)

Volume. This register set controls the volume level of the sound created.

Address: \$40-\$5F

Waveform data sample. This reads the last value from the waveform table.

Address: \$60-\$7F

Address pointer. These registers are used to determine where in RAM the waveform tables are located. Each waveform table begins with the first address of a page and must continue upward through RAM and cannot wrap around over 64K. The register keeps track of where the table ends.

Address: \$80-\$9F

Control register. Channel assignment, oscillator mode, and halt bit are all controlled by this register. Bits 4-7 make up the channel assignment. Those four bits can assign up to 16 channels for sound. Bit 3 is the interrupt enable used for ordering output when more than a single oscillator has generated output. It helps keep all the different sounds organized. Bits 1 and 2 set the oscillating mode for each oscillator, and bit 0 is the halt bit indicating when an oscillator has been stopped by the microprocessor or DOC.

Address: \$A0-\$BF

Bank select/resolution/waveform registers. Each register uses seven bits for controlling three major functions (bit 7 is not used). Bit 6 determines whether the DOC address range is 0-64K (0) or 65-128K (1). Bits 3-5 specify the size of the waveform table, ranging from 256 bytes to 32K bytes. Finally, bits 0-2, called the *resolution determination bits*, actually determine the final address for the waveform table.

Address: \$C0-\$DF

Table 3: Drawing Calls

Lines

LineTo Line

Rectangles

FrameRect PaintRect EraseRect InvertRect FillRect

Regions

FrameRgn PaintRgn EraseRgn InvertRgn FillRgn

Polygons

FramePoly PaintPoly ErasePoly InvertPoly FillPoly

Ovals

FrameOval PaintOval EraseOval InvertOval FillOval

Rounded-Corner Rectangles

FrameRRect PaintRRect EraseRRect InvertRRect FillRRect

Arcs

FrameArc PaintArc EraseArc InvertArc FillArc

Pixel Transfers

ScrollRect PaintPixels PPToPort

Text Drawing and Measuring

DrawChar DrawText DrawString DrawCString CharWidth
TextWidth StringWidth CStringLength CharBounds TextBounds
StringBounds CStringLength

Mapping and Scaling Utilities

MapPt MapRect MapRgn MapPoly ScalePt

Miscellaneous Utilities

Rectangle Calculations

SetRect OffsetRect InsetRect SectRect UnionRect
PtInRect Pt2Rect EqualRect EmptyRect

Point Calculations

AddPt SubPt SetPt EqualPt
LocalToGlobal
GlobalToLocal

Region Calculations

NewRgn DisposeRgn CopyRgn SetEmptyRgn
SetRectRgn RectRgn OpenRgn CloseRgn
OffsetRgn
InsetRgn SectRgn UnionRgn DiffRgn
XorRgn
PtInRgn RectInRgn EqualRgn EmptyRgn

Polygon Calculations

OpenPoly ClosePoly KillPoly OffsetPoly

Other

Random SetRandSeed GetPixel

Oscillator interrupt, oscillator enable, and A/D converter registers. These three registers (not bits) control the oscillators and analog-to-digital conversion.

Address: \$E0-\$E2

Sound Tools

From the above brief description of the sound registers and the digital oscillator chip, you can see that it's not simple to crank up the kinds of sound heard in musical demonstrations on the IIGS. As an aid to programmers, the sound tools have been provided. There are 18 sound function calls and six low-level routines for accessing the power of DOC. The sound toolkit works through a sound tool set with a specified number. The tool locator finds this number in order to use the sound tools. Again, this requires a higher level of programming skills than you may now possess, but to give you an idea of what's in the sound tool set, the calls in Table 4 are available.

Table 4: Sound Tool Calls

Function Calls

SoundBootInit	SoundStartup
SoundShutdonw	SoundVersion
SoundReset	SoundToolStatus
WriteRamBlock	ReadRamBlock
GetTableAddress	GetSoundVolume
SetSoundVolume	FFStartSound
FFStopSound	FFSoundStatus
FFGeneratorStatus	SetSoundMIRQV
SetUserSoundIRQV	FFSoundDoneStatus

Low-Level Routines

Read Register	Write Register
Read Ram	Write Ram
Read Next	Write Next

For some of you, it may be frustrating not to be able to program sound and super high-resolution graphics on your IIGS with what you now know about programming, but be patient. You will learn the more advanced techniques in time. Books and programming utilities will be available in the future to help you. ©

COMPUTE!

Label Master

This Commodore 64 program from the February issue (p. 56) and on the COMPUTE! Disk for February-April works properly on very old 64s with version 1 ROMs and on newer 64s with version 3 ROMs—including the 64C and the 128 in 64 mode. However, the graphic-design grid doesn't appear on older 64s with version 2 ROMs. To check which ROM version your 64 has, enter PRINT PEEK(65408). If the value returned is 0, your ROMs are version 2. (A value of 170 indicates version 1, and a value of 3 indicates version 3.)

When the screen is cleared, version 2 fills color memory with the screen background color. Since "Label Master" makes no provision for changing color memory, characters stored into screen memory are the same color as the background, and hence are invisible. The following patches modify this program to work with version 2 ROMs. To make the changes, first load a complete copy of Label Master into "MLX" using the starting and ending addresses given in the article. Next, select the (E)nter data option, specify an address of 0801, and enter the following lines:

```
0801:0B 08 0A 00 9E 32 30 37 2F
0809:37 00 00 00 A9 01 8D 21 42
0811:D0 A9 93 20 D2 FF A9 00 52
0819:8D 21 D0 60 A9 02 85 FD B6
0821:8D 20 D0 A9 50 85 FB A9 EE
0829:04 85 FC 18 20 0D 08 A0 A3
```

Then change the entry address to 09E9 and enter this line:

```
09E9:00 20 0D 08 A0 00 B9 FD 9C
```

Change the entry address to 0AF1 and enter this line:

```
0AF1:00 20 0D 08 20 7B 0B A9 DE
```

Finally, save a copy of the modified Label Master data before exiting from MLX.

Menu Planner For The Commodore 64

The 64 version of this program from the April 1987 issue (p. 60) has a number of mistakes in the routine which sends menu lists to a printer. To properly print menus, change the following lines:

```
610 PRINT"MENU:":IFA$="P"THENO
    PEN4,4:CMD4:PRINTSPC(45)"ON
    HAND{4 SPACES}NEED TO BUY"
640 IFA$="R"THEN GETA$:IFA$=""
    THEN640
660 IFA$="P"THENPRINT#4:CLOSE4
```

Superplotter

Line 940 of this Commodore 64 graphics program in the March 1987 issue (p. 120) includes the delete ({DEL}) character, which can't be typed in the usual way. To produce this character, use the cursor keys to move the cursor to the spot in the line where the {DEL} is to be inserted, use the SHIFT-INST/DEL key combination to insert a space, and then press INST/DEL again (without SHIFT). The delete character will appear as a reverse-video T. ©

Back Issues of
COMPUTE!
COMPUTE!'s Gazette,
or any magazine disks
can be ordered by
calling **800-346-6767**
(in NY 212-887-8525).
Some issues may no
longer be available.



A Nation Of Thieves: Responses To Readers

Last October I wrote what, in my view anyway, was a fairly calmly written piece on the illicit copying of software. I mentioned the impact that illicit copying had on software companies and on those of us who choose to obey federal law on this topic. Based on letters I received, one would have thought that I was attacking motherhood!

I knew I was in trouble the day I went to my post office box and all I saw was a pink card instructing me to pick up my mail at the counter. Over the weeks following the publication of that article, my accumulated mail on this topic filled a #3 mailbag. People wrote from as far away as Sweden to challenge my position.

Of the letters received, only two readers wrote to thank me for sensitizing them to a major industry problem—and, no, these people were not mouthpieces for the software companies. I decided to read each letter. Some readers got sidetracked from the issue and attacked me personally, at which point their letters were (generally) discarded.

I placed the remainder of the letters in several piles based on the general themes of the arguments presented. The remainder of this article summarizes the main arguments against my position and includes my responses.

Letter writers are identified only by initials for two reasons. First, some of you admitted to acts that are punishable by incarceration in a federal penitentiary, and I felt it to be in your best interest if I excerpted what I needed from your letter and shredded the rest. Second, there was not a great deal of novelty in the responses, so any point one reader might have made was also made by others.

I also have taken the liberty to change some of the language of the arguments so they can be printed in

a family magazine.

The "Service To Society" Argument

"Why is it that software publishers seem to think that they are any different than the guy who writes some worthwhile book? For centuries the world has known that authors are generally underpaid and underappreciated, and yet there is no great outcry in their favor.

Who has more intellectual power invested: The guy who writes a straightforward spreadsheet such as Lotus 1-2-3 or the guy who writes a book such as War and Peace? I would suggest it is the latter."—DB

Well, DB, everyone I know who writes a worthwhile book expects to get paid for his or her efforts, and usually does. Yes, history is full of examples of authors who didn't get rich from their efforts, but there are plenty that do. I know a science fiction author who gets \$1,000,000 advances for his books, and his case is not that uncommon. If you write a book that becomes a best seller, you bring in big bucks, period. Fortunately, books are expensive to copy—but programs aren't, so software gets ripped off.

Your second question has a very simple answer, the "guy" (actually an entire company) that wrote Lotus 1-2-3 has a lot more energy invested than Tolstoy ever put into *War and Peace*. For example, Tolstoy's hand-written drafts had lots of mistakes, some of which made it into the published version of the book. The reviews of his work never mentioned the misspelled word on page 315 or the awkward phrase on page 74. "Bugs" of this nature cause havoc in a computer program, but are perfectly acceptable in a book. The consequences of typing errors in books is benign—you may wince (depending on your fluency in Rus-

sian), but you forgive Tolstoy all the same.

This is not the case with computer software. A typing error may result in a subtle bug that takes weeks to fix. In the meantime, the users keep complaining and requesting a free upgrade once the bugs are fixed. I agree that free upgrades are appropriate for software, and I do not expect publishers to send revised editions of books to their old customers at no cost—even though the parallel that some readers imply would require such service.

No, the point is that the publishing of software is quite different from the publishing of books. I know this because I do both, and books, comparatively, are a snap to publish!

Let's pretend that Tolstoy had to follow the path of a software publisher when he wrote his epic novel. First, he would have had to create several releases of his book to repair his "bugs." Then, he would have to immediately start working on his next book, knowing that if he didn't improve on his product, others would. Most of his profits would have to be placed into this new venture.

Compared to software authors, book authors have it made.

The "Mouthpiece For Big Business" Argument

Now that you former hackers have hit the big time, you've developed vested interests in protecting the vendors to the detriment of the consumers. The real thieves are the companies that can charge outrageous prices for programs that are guaranteed not to work. What other industry do you know that will sell a product and not make full refunds on defective merchandise?

Your blanket statement that copying software is illegal is tanta-

mount to saying that no one has the right to free speech.—WM

You bet we have vested interests, just like the kid who was willing to play sandlot ball for free, but who now gets hundreds of thousands of dollars a year as a pro. I'm sorry, but this argument just doesn't work. I've never seen a software company put out a product guaranteed not to work; nor have I encountered a company that was unwilling to refund a customer's money when the product was returned in a timely manner with documentation to support its return. I say this even though there are some people who will buy a product, make a copy, and then return the product because "it doesn't meet their needs."

As for my blanket statement, I was very careful to level my criticism at those who copy to secure software that they haven't purchased. Backup copies, media conversion, and so forth, are all within the domain of reasonable copying, and do not (to my knowledge) violate any laws.

The "Software Is Lousy, So Why Buy It?" Argument

The list of lousy software that readers came up with was quite large. A typical comment went like this:

You bet I have an illegal copy of XXX; it is a piece of trash. For example, every time I boot the program I have to enter a bunch of preferences all over again. Why didn't the developer let you save your preferences on disk? It wouldn't matter to me if I used the program infrequently, but I use it every day and this is a pain in the tail.

This comment—synthesized from several letters—presents an interesting definition of trash. Every morning I empty the trash so it won't smell up the house. Trash is something you throw away, not something you "use every day." The "trash" argument is just a thinly veiled disguise for the feeling of guilt that these readers must be experiencing.

By the way, "legal" users of your programs have probably received upgrades that fix the problems you mentioned in your letters.

The Real Issue

Behind the anger expressed in many of your letters lies a real issue that needs to be resolved. Many of you feel that software vendors aren't responsive to your legitimate needs. The products don't live up to expectations, the documentation is full of errors, the product's function doesn't justify its cost, and so on. All of these are legitimate complaints that should be addressed to the software publishers. By communicating with these people directly, you stand an excellent chance of seeing your ideas incorporated in new releases of the product. I always send free copies of my programs to people whose suggestions I have incorporated, and this is not an uncommon practice in this industry.

If you feel that the software vendors aren't listening, maybe this is because you aren't speaking to them. By sneaking around behind the vendor's back and stealing software, you are losing any chance you ever had to make sure that the programs you really want will ever make it to the marketplace.

Dr. Thornburg's most recent product is Calliope™, a "non-linear" idea processor for the Apple IIe, c, GS, and Macintosh computers. He welcomes letters from readers and can be reached in care of this magazine. ©

COMPUTE!'s Gazette is looking for utilities, games, applications, educational programs, and tutorial articles. If you've created a program that you think other readers might enjoy or find useful, send it, on tape or disk, to:

**Submissions Reviewer
COMPUTE! Publications
P.O. Box 5406
Greensboro, NC 27403**

Please enclose an SASE if you wish to have the materials returned.

Articles are reviewed within four weeks of submission.

Train for the Fastest Growing Job Skill in America

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

Now you get it all... training for America's fastest growing career opportunity... training to service all computers... training on the newest total computer system, the Sanyo 880, yours to keep.

Get inside the newest, fully IBM-PC compatible Sanyo Microcomputer

As an NRI student, you'll get total hands-on training as you actually build the latest model Sanyo 880 Series computer from the keyboard up. It's fully IBM PC compatible and, best of all, it runs programs almost twice as fast as an IBM PC.

As you assemble the Sanyo 880, you'll perform demonstrations and experiments that will give you a total mastery of computer operation and servicing techniques. You'll do programming in BASIC language—even run and interpret essential diagnostic software.

Learn at home in your spare time

You train in your own home at your convenience, backed at all times by your NRI instructor and the entire NRI staff of educators and support people. They're always ready to answer your questions and to give you guidance and special help wherever you need it.

100-page, free catalog tells more... send today

Send the coupon today for NRI's 100-page catalog that gives all the facts about computer training plus career training in other electronics fields.



Send Coupon Today for Free Catalog!

NRI School of Electronics

McGraw-Hill Continuing Education Center
3939 Wisconsin Ave., Washington, D.C. 20016

Name (Please Print) Age

Address

City/State/Zip

198-067



Lapping It Up

The face of mobile telecomputing is changing. I've owned a Tandy 100 laptop for several years. When first introduced, the Model 100 significantly expanded the freedom of a generation of journalists. Its portability and built-in 300-bps modem made it possible to write and transmit a story on the spot. A new wave of battery-powered PC-compatible laptop machines—which overcome the small display and memory limitations of the Model 100 with full-sized screens and 640K of memory—is swelling the ranks of mobile telecomputers. If you've been thinking of joining them, here's some friendly advice.

Most PC laptops offer 300-/1200-bps internal modems as an option. They are somewhat pricey, listing for \$400 or so, and place an additional drain on the laptop's batteries, shortening the amount of usable time between charges. Compact and attractively priced, battery-powered modems are viable alternatives to internal laptop units. Most notable at this time are Touchbase System's WorldPort 1200 (\$199) and Migent's Pocket Modem (\$259). Both are compatible with the Hayes command set; about the size of a pack of cigarettes, they plug directly into any computer's serial port. They can be powered from an internal battery when you're on the go, or from an AC adapter when an electrical outlet is handy. A pocket modem is a good choice if you are on a limited budget or want to use your modem with more than one computer. If you're completely immersed in the portability gestalt, an internal modem means one less item to lug around.

When I'm on the run, treating work gear gingerly is usually the last thing on my mind. Since unceremonious dunks and tosses into car trunks are a fact of life for telecomputing on the hoof, a suitable carry-

ing case is a must. Most of the carrying cases offered by the laptop manufacturers themselves offer little real protection and stubbornly refuse to acknowledge the fact that you may want to carry anything other than your computer. More often than not, well-padded cases designed for photographic equipment can be pressed into laptop service with little or no compromise. Fans of "Miami Vice" will opt for the polished aluminum, high-tech look of Halliburton Zero cases. Since my trusty Volvo is decidedly less flashy than a Ferrari, I went for a sedate nylon LowePro Trimtech photographer's case that has enough padding to insure survivability in environments short of a direct nuclear strike; plenty of room for accessories, business papers, and manuals; and a fold-down "office" for storing disks, writing implements, business cards, and other goodies.

You Can't Get There From Here

Then there's the problem of moving data between your laptop and desk machine. Most desktop PCs are equipped with 5¼-inch disks, while the current breed of laptops use 3½-inch microdisks. In order to bridge the gap, many laptops offer external 5¼-inch disk drives as optional accessories, and a 3½-inch disk drive can be added to almost any PC compatible. Adding an internal microdisk to your PC is not too expensive (about \$200), but limits your data transfers to wherever your desktop computer sits. An external drive for a laptop is more expensive (about \$400) and saddles you with another relatively bulky piece of equipment to lug around. Remember, in the laptop world, as in Bauhaus architecture and California politics, "Less is more."

The humble standard RS-232

serial port found on almost all PCs offers efficient, reasonably priced alternatives for laptop-to-PC file transfers. Using standard public domain or "shareware" telecommunications software and a null modem cable lets you move files between machines at speeds of up to 19,200 bits per second. White Crane System's *Brooklyn Bridge* software (\$129.95) makes file movement between laptop and desktop PCs even more of a snap. *Bridge* lets either machine access the other's drives directly, using standard DOS commands. The special drivers supplied allow data to move between systems at 115,000 bps, which is pretty close to the maximum transfer speed rate of laptop microdisk drives.

Padded Cells

Where will this mobile telecomputing madness end? Within the next few weeks, I'll be shopping for a cellular mobile phone that will allow hookup to my NEC Multi-Speed's internal modem. Will life ever be the same? Will online teleconferencing and stock quotations replace such classic highway games as "Geography," "License Plate Lotto," and endless choruses of a "Hundred Bottles of Beer on the Wall?" Stay tuned. ©



Microscope

Sheldon Leemon

In the music industry, compact discs are riding high. When the shiny silver platters were introduced a couple of years ago, players cost over \$1,000, and there were only a small number of albums to play on them. Now it's possible to buy CD players for as little as \$100, and a vast library of music is available on CDs.

CDs apply digital technology to the recording of music. Sounds are translated into binary numbers composed of 0's and 1's, and stored as a series of high and low spots on a disc that can be read by a laser beam. The same digital principles are used by microcomputers, which translate data such as text characters into a series of binary numbers. So CDs seem an even more natural medium for storing computer data than for storing music.

So far, Microsoft, the software giant, has lead the way in promoting standards for using CDs as computer storage devices. This technology is known as CD-ROM (Compact Disc-Read Only Memory), though CROD (Compact Read Only Disc) would be more accurate. Each tiny removable disc holds about 500 megabytes of data, enough to store over 100,000 pages of text. Since discs are stamped out like records, that vast quantity of information can be duplicated in a fraction of a second, at a cost comparable to that of floppy disks. Even though there's currently no way to erase and write over a CD as you can with a floppy disk, the computer industry sees CD-ROMs as an inexpensive way to provide personal computers with access to vast reference libraries. It's possible to put all of the volumes of an encyclopedia on one CD, along with an index of every word used, so that every occurrence of a word or phrase can be looked up in seconds. In fact, the text of the ency-

clopedias would only take up a part of the space available, leaving a lot of room for other books as well.

In March, Microsoft held its second annual CD-ROM Conference. Though there are still almost no commercial CD-ROM products available (particularly at the consumer level), there were over 1200 attendees and exhibitors at the conference, which indicates the level of interest in this technology. At the conference, Microsoft unveiled *Bookshelf*, a \$295 package made up of a memory-resident IBM PC program and a CD containing ten reference works. Included are a dictionary, a thesaurus, a style guide, *Bartlett's Quotations*, a 1987 almanac, and a ZIP code directory. The PC program is designed to work with any word processor. You could, for example, use it to look up a ZIP code while writing a business letter, and have it insert the ZIP code into the text of the letter at the appropriate place. Microsoft also announced at the conference that it would bundle *Bookshelf* with Amdek's Laserdrive I, an external CD-ROM player that can also play audio disks. The \$1,099 package includes a PC drive controller and Microsoft's MS-DOS CD-ROM extensions, which make it possible for PC programs to read data from CDs much as they do from ordinary disks.

So far, the applications for CD-ROM technology have been very specialized and expensive. Lotus has been offering a financial database called *One Source*, for example, that contains detailed financial information about publicly held companies, and has announced that it will update that information weekly (for a fat fee). It's also possible to get *Books in Print*, the standard reference for booksellers, on a single CD, at a cost of about \$800. Information vendors feel that they

must charge such high prices because there's a relatively small number of CD-ROM owners to whom they can sell their wares. But until there's more reasonably priced software available, few CD-ROM players will be sold. By bundling *Bookshelf* with players, Microsoft hopes to get the ball rolling on both the hardware and software side.

As exciting as CD-ROM is, it's not the only computer-controlled CD technology. Standards have already been established for an interactive audio disc technology called CD-I (Compact Disc-Interactive). And at the Microsoft conference, RCA put on a demonstration of what it calls DVI (Digital Video Interactive). This new technology packs as much as 72 minutes of high-quality full-motion video and audio on a single disc. The maximum video resolution of this system is an excellent 768 × 512 pixels, with up to 16 million colors per pixel. Such interactive video technology could be used for educational materials, videogames, and a host of applications that have been barely dreamed of as yet.

So far, CD-ROM has not visibly benefited from the success of its audio counterpart. Since CD-ROM requires much higher accuracy than music CD players afford, the plummeting price of music CD players has not brought a corresponding drop in the price of CD-ROMs. Atari has been forced to delay plans for a \$500 player for its ST computer because the company has been unable to meet the target price. But the spread of music CD technology, and especially the expansion of disc-production capacity, should work to the advantage of CD-ROM in the long run. ©



Instant Images On Your Apple Computer

As part of our Multi-Media Classrooms Project here in Birmingham, Alabama, I've had the opportunity to play with an exciting new product, the FingerPrint Plus Card from Thirdware Computer Products (4747 N.W. 72nd Ave., Miami, FL 33166). This product has a unique appendage—a little ribbon cable and an activator tab with the picture of a fingerprint on it. The tab projects from the front of your Apple IIe, II+, II, or IIGS computer. When you press it, you have an instant snapshot of your current computer screen.

The FingerPrint Plus card (with installation, slide-show software, and activator tab) normally retails for \$149. However, schools get a 35 percent discount. And, if it's the first FingerPrint Plus for a school, it costs only \$75.

When I received the card from Thirdware, I installed it in slot #1 in my Apple IIe. The card comes with interfaces for parallel and serial printers, but it's a good idea to make sure you have the right cable (parallel or serial) for your printer. I tested the card on several brands of software, including *Bank Street Writer*, *Print Shop*, *Apple Access*, and *Magic Slate*, and I had no printing problems.

Looking at the documentation, I learned that the card can also double as a serial modem card. All you need is a special FingerPrint Plus-to-modem cable, and you need to make some dip-switch changes. The FingerPrint Card also can turn your Apple into a fancy typewriter. Choosing the Type option on the main menu screen allows you to type directly onto your printer.

Easy As Pie

Capturing a computer screen is easy. You never have to leave your software or issue any special commands. When you see a screen you like—even if it's visible momentari-

ly—you press the fingerprint tab. The screen vanishes, and in its place is a full-screen menu of FingerPrint Plus functions. At this point, if you press the Return key, a copy of the screen is sent to your printer. Or you can choose Display and take another peek at the screen to make sure it's the one you wanted. When you return to the menu (by pressing the fingerprint tab), you can choose Disk on the menu and save your screen image to disk.

After you've saved several screen images to disk, you can boot up the FingerPrint Plus disk and create a custom-made slide show. You can present the slides in any order and select the time each slide remains on the screen during the show (a minimum of ten seconds per slide).

Back in the main menu, there are several other things you can do to manipulate your captured screen image. You can select the number of copies of the image you want printed. Page numbers and headings can be inserted on the pages. You can invert the computer's screen image and print the inverted image. You can enlarge the image to twice its normal size, rotate it 90 degrees, or select just a portion of the image for printing. Finally, you can manipulate the colors in your image when it is printed out.

Some minor caveats: When I got my first FingerPrint Plus card, it caused my Imagewriter II to mysteriously jump columns when printing hi-res screens. A second card suffered from the same problem. I suspect that the problem was caused by the Imagewriter II's 32K memory option, because when I tested the same cards on Imagewriters without the option, the card worked perfectly.

Also, when I created my first slide show, the screen cursor behaved oddly and began scrolling

my menu of slides off the screen. However, when I took the COMPUTEREYES/2 video digitizer card out of slot 3 in my Apple IIe and put it into slot 7, the cursor again became well-behaved.

Many Applications

It's a lot of fun to press the fingerprint tab and watch a computer screen instantly print out. But what are some practical uses for a FingerPrint Plus?

- Create custom-made quick-start cards and training manuals for your students or other new software users. You can pick key screens in the software, take FingerPrint snapshots, and use the screens to illustrate your training instructions.
- Create activity worksheets for your students. Lots of educational programs have graphics screens which illustrate math, science, social studies, and language arts concepts. You can capture the screens, duplicate them, and use them as worksheets for your students to do at their desks.
- Capture key screens, organize a slide show, and either present the show on a large-screen monitor or video projector, or send your computer's video output to a VCR and capture it on videotape. You'd then have a video outline illustrating key parts of your software or illustrations for your presentation on a given subject.
- Print out black-and-white screen dumps onto transparencies on your printer (using a carbon plastic ribbon) and show the transparencies with an overhead projector.
- Using a COMPUTEREYES/2 video digitizer (\$129 from Digital Vision) and a FingerPrint Plus card, digitize live video images and images from books, magazines, and so on. Then create a slide show of images and send it via modem to other schools. ©



Journey From The Center Of The ST: Part 1

Explain, in 25 words or less, each of the following ST acronyms: BIOS, XBIOS, GEMDOS, TOS, VDI, GDOS, AES, GEM. Nearly every ST owner knows that these terms identify various parts of the ST's system software—the fundamental “stuff” that makes it work. But what does each one do, and how do they all fit together? We'll spend the next two months seeking basic answers to these questions.

This knowledge is essential for any serious programming. And even if you're not aspiring to guru status, you may find the survey helpful for those occasions when the air fills with a thick fog of computerspeak. We'll start at the center—the most primitive level of the machine—and work our way out. Waiting for us when we get to the surface will be GEM, the highest level of system software and the element responsible for much of the ST's personality.

More Than A Pile Of Chips

Although you may picture computers as mechanical devices, the most important part of any computer consists of programs. The hardware inside a computer—memory chips, microprocessor, and so on—differs little from one machine to the next and is almost useless by itself. What brings the hardware to life is the system software, a collection of built-in programs which the computer uses to function.

The most basic level of system software is the operating system. Some machines, like the current ST, store this software in a ROM (Read Only Memory) chip. Others, like the original ST models, load the operating system from a floppy disk when you turn on the power. No matter where it comes from, the operating system is necessary in order for the computer to do anything at all.

Layer Upon Layer

The ST's operating system, collectively dubbed TOS, is actually composed of three separate parts: BIOS (Basic Input/Output System), XBIOS (eXtended BIOS), and GEMDOS (GEM Disk Operating System).

If you visualize the ST's system software as a series of layers, then BIOS is the lowest layer. The BIOS makes it possible for the ST to perform simple, fundamental tasks which are necessary and common to every modern computer: printing a character, writing to a disk sector, and so on. Without this base level of functionality, none of the computer's higher capabilities would be of use.

The next layer of the operating system is XBIOS, which provides more sophisticated input/output functions and supports hardware features that the ST doesn't share with other computers. The XBIOS lets you do tasks such as changing the color palette, communicating through the MIDI port, reserving blocks of memory, or formatting a floppy disk. XBIOS is a bit of a grab bag, but its functions generally involve some low-level task tied to an ST-specific architectural or hardware feature.

Next we come to GEMDOS, which provides high-level disk operations such as searching a directory, as well as more sophisticated character input/output, large-scale memory management, and functions for loading and running programs. While BIOS and XBIOS are tied to hardware characteristics, GEMDOS is said to be hardware independent. That is, GEMDOS calls BIOS and XBIOS routines to perform hardware functions, rather than directly accessing the ST's hardware.

This hierarchical arrangement typifies the system software as a whole. As with a ladder or pyramid, the higher levels of the structure

depend on lower levels for support. At the bottom are fundamental routines which perform one or two rudimentary tasks and are comparatively simple to use. As you move up the ladder, system routines become more powerful and versatile, and often more complicated to use. It takes a lot more setup to, say, open a GEM window, than it does to output a single character. And you have a lot more decisions to make when calling the open-window routine.

The Face Behind The Mask

Once we install the three software elements of its operating system—BIOS, XBIOS, and GEMDOS—the ST becomes a working device. If the Atari designers had stopped with TOS, the ST would be something like a fast MS-DOS computer with lots of memory. You would have a keyboard, simple character output to the screen, and the ability to access peripherals such as the disk drive.

In fact, there's a simple way to use the PC-like machine hidden inside the ST. When you run a GEM application (a file that ends with .PRG), you have the complete computer at your disposal, including its flashy and convenient visual interface. But when you run a TOS application (a file that ends with .TOS or .TTP), the ST strips off the glamorous mask of GEM and uses only the operating system. The TOS-only machine is powerful, but rather dull and cumbersome compared to the ST we use most of the time.

What's missing from this picture, of course, is GEM—the user interface that provides the windows, menus, and other graphics elements that make the ST much more than just a fast PC. We'll finish our journey in the second part of this column, which examines the various software elements that make up GEM. ©



The Protection Racket

How many times have you written a program that is supposed to write to a disk file, only to have the drive make an ugly grinding noise and then have BASIC (or your program, if it is doing error trapping) tell you that the disk is write-protected? If you're like me, the answer is *very* often.

I'm usually cautious when writing my programs. I leave the write-protect tab on all my disks until I am 99 percent certain that the program will work. So during the development of a program, I tend to get lots of error number 144 messages. (Error 144 can mean anything from a too-fast disk drive to a 1050 drive's door being open, but most often it means that a disk is write-protected.)

Testing First

Wouldn't it be neat to be able to test if a disk is write-protected *before* you open a file for a write? Actually, you can. The method is a fairly obscure one; and to find it, once again I had to consult the old *Atari Technical Reference Manual*.

A short sidetrack: Those of you who don't mind searching through a couple hundred pages to find a small item buried in a lot of software (and hardware) engineering talk really should own a copy of this manual. It is now a six-year-old document, but it is still useful and accurate (aside from some of the RAM locations used by the newer XL/XE operating systems—for which you can consult COMPUTE!'s revised version of *Mapping the Atari*). This is a tribute to the design capabilities of the engineers from the old Atari and the astuteness of today's Atari: Never has an entire line of personal computers stayed so compatible.

Where The Secret Lies

The secret to the write-protect is

buried in the information about the disk drive's status command. (Don't confuse the drive status with a file's status, as tested by BASIC's STATUS command.) If you can recall my September 1985 column regarding SIO (Serial Input/Output) commands, it may be easier for you to understand the code which follows. I am not going into great detail, but, briefly, an SIO call must have certain information placed in page 3 (locations \$300 to \$30B, specifically) before the program jumps to the subroutine at location \$E459. (You can use my handy-dandy number converter program of a few months back to convert those hex numbers to decimal.) Information placed in page 3 includes the drive number, type of request (S, for status, in this case), the address of a buffer, and the number of bytes to transfer.

It is this last set of information that is most important to us: The drive status returns four bytes of information that we need to receive somewhere so that we can analyze it. We'll take a closer look at the accompanying listing later, but for now just notice that we dimension a buffer (BUF\$) in line 30100. The four bytes of BUF\$ will be used to receive the four bytes of drive status. We'll be using only the first byte of that status, because that is where the write-protect flag is located. Some of the bits in that byte are related to various hardware error conditions, which we need not discuss, leaving the following bits as useful to us:

- Bit 3 \$08 write-protected disk
- Bit 5 \$20 double density disk
- Bit 7 \$80 1050 enhanced density disk

Well, well! So not only can we find out if the disk is write-protected, we can also find its density.

A Useful Subroutine

In this month's listing, then, lines

30000 and beyond are a subroutine that you can include in your own programs. To use the subroutine, simply set the variable DRIVE to a valid drive number (1 to 4, usually) and GOSUB 30000. Upon return, the variable CHECK will contain one of the following values:

- less than 0 an error occurred (invalid drive number, for example)
- 0 disk may be written to
- 1 disk is write-protected

If the value returned is less than zero, then the value will be the negative of the appropriate error code. For example, if you try to check a drive that isn't turned on, the value returned should be -138, indicating that error 138—device timeout—has occurred.

Similarly, a second number is returned in a variable named DENSITY. Its meaning:

- 1 density value was invalid
- 1 drive is single density
- 2 drive is double density
- 3 drive is enhanced density

And all of this is demonstrated by the code in lines 100 through 220. These lines are provided just to give you a test bed to try out the subroutine of lines 30000 and up. Try the program, and then incorporate the subroutine in your own programs. And never again will you or your users see that dreaded error number 144 (unless your drive speed is off—but that's another topic).

Just a couple of last comments: Notice line 30170. This demonstrates another programming trick. Worried about making sure that your RESTORE statements refer to the right DATA line numbers? For short DATA lines, why not simply combine RESTORE and DATA on the same line, as in 30170?

Also, notice the sneaky way that BUF\$ is dimensioned in line 30100. If we come to line 30100 a second time, we get an error when we try to re-DIMension BUF\$. But,

because of the TRAP, the error is effectively ignored—a useful way of making sure a variable is dimensioned only once.

Write-Protect Checker

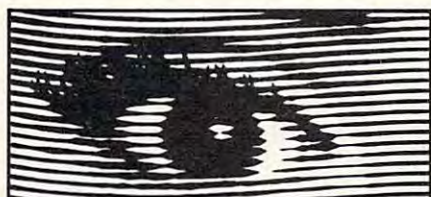
For instructions on entering this program, please refer to "COMPUTE!'s Guide to Typing In Programs" elsewhere in this issue.

```
EA 100 REM SIMPLE PROGRAM TO
DEMONSTRATE
BC 110 REM THE WRITE PROTECT
CHECKER ROUTINE
HA 120 PRINT :PRINT "
DRIVE NUMBER ";
JD 130 INPUT DRIVE
AJ 140 GOSUB 30100
KG 150 IF CHECK<0 THEN PRINT
"ERROR # ";-CHECK;"
ACCESSING DRIVE ";DRIVE:GOTO 120
CJ 160 IF CHECK>0 THEN PRINT
"DISK IS WRITE PROTE
CTED";
LI 170 IF CHECK=0 THEN PRINT
"DRIVE IS READY";
PC 180 IF DENSITY=1 THEN PRI
NT ", SINGLE DENSITY.
"
ON 190 IF DENSITY=2 THEN PRI
NT ", DOUBLE DENSITY.
"
BB 200 IF DENSITY=3 THEN PRI
NT ", ENHANCED DENSIT
Y."
FI 210 IF DENSITY<0 THEN PRI
NT ", UNKNOWN DENSITY
."
```

```
FD 220 GOTO 100
KG 30000 REM SUBROUTINE TO C
HECK DISK FOR WRITE
PROTECT
NI 30010 REM
HM 30020 REM ENTER WITH DRIV
E TO CHECK IN DRIVE
(1 TO 8)
NK 30030 REM
JJ 30040 REM ON EXIT, CHECK
WILL NORMALLY BE 0
LD 30050 REM IF CHECK IS 1,
DISK IS WRITE PROTE
CTED
HD 30060 REM IF CHECK IS <0
THEN CHECK IS NEGAT
IVE OF SIO ERROR CO
DE
ON 30070 REM FOR CHECK=1 OR
0, DENSITY IS RETUR
NED:
JL 30080 REM DENSITY IS 1,2,
3 FOR SINGLE, DOUBL
E, ENHANCED DENSITY
BB 30090 REM DENSITY IS -1 I
F UNRECOGNIZED
LK 30100 TRAP 30110:DIM BUF$
(4)
BH 30110 POKE 768,49:POKE 76
9,DRIVE
PI 30120 POKE 770,83:POKE 77
1,64
CE 30130 POKE 773,INT(ADR(BU
F$)/256)
CH 30140 POKE 772,ADR(BUF$)-
256*PEEK(773)
JA 30150 POKE 774,2:POKE 775
,0
JH 30160 POKE 776,4:POKE 777
,0
```

```
AB 30170 RESTORE 30170:DATA
104,76,89,228
BA 30180 FOR BYTE=1 TO 4:REA
D CHECK
BN 30190 BUF$(BYTE)=CHR$(CHE
CK):NEXT BYTE
NK 30200 BYTE=USR(ADR(BUF$))
BE 30210 CHECK=PEEK(771):IF
CHECK>127 THEN CHEC
K=-CHECK:RETURN
DJ 30220 CHECK=0:BYTE=INT(AS
C(BUF$)/8)
GC 30230 IF BYTE/2<>INT(BYTE
/2) THEN CHECK=1
DC 30240 BYTE=INT(BYTE/4):DE
NSITY=-1
OH 30250 IF BYTE=0 THEN DENS
ITY=1
OK 30260 IF BYTE=1 THEN DENS
ITY=2
OP 30270 IF BYTE=4 THEN DENS
ITY=3
NN 30280 RETURN ©
```

COMPUTE!'s GAZETTE
TOLL FREE
Subscription Order Line
1-800-247-5470
In IA 1-800-532-1272



Looking for Thermal Paper or Mailing Labels for your Okimates? Call Precision!

Precision Images now has available for your Okimate printers, GENUINE Okidata thermal transfer roll paper and mailing labels. We also carry a large supply of spare parts and supplies for all Okidata printers. Precision Images is "your direct connection to genuine Okidata parts and supplies."

New Microline 93 Printer—\$375

for Visa/MasterCard orders call:
1-800-524-8338



Precision Images, Inc.
P.O. Box 563, Dept. C
Chester, New York 10918

Attention all FX80, FX100, JX, RX, & MX owners:

You already own half of a great printer

Now Only \$79.95



Now for \$79.95 you can own the rest. You see, today's new dot matrix printers offer a lot more.

Like an NLQ mode that makes their letters print almost as sharp as a daisy wheel. And mode switching at the touch of a button in over 160 styles. But now, a Dots-Perfect

upgrade kit will make your printer work like the new models in minutes—at a fraction of their cost.

And FX, JX and MX models will print the IBM character set, too.

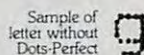
So, call now and use your Visa, MasterCard, or AmerEx. Don't replace your printer, upgrade it!

1-800-368-7737

(Anywhere in the United States or Canada)



Dots-Perfect™



Dresselhaus

8560 Vineyard Ave., Ste. 405, Rancho Cucamonga, CA 91730

(714) 945-5600

An upgrade kit for EPSON FX, JX, RX, & MX printers

IBM is a registered trademark of International Business Machines Corp. Graphics™ is a trademark of Epson America, Inc. Epson is a registered trademark of Epson America, Inc.



The Beginner's Page

C. Regena

Arithmetic In BASIC

In most of my guest lectures or computer presentations, I emphasize that a mathematics background is *not* necessary to use or to program a computer. However, we do use numbers quite frequently in programming. This month we'll discuss a few arithmetic operations that are important when programming in BASIC.

The computer is a sophisticated calculator, and you can add and subtract numbers by using plus and minus signs:

```
PRINT 4 + 3
PRINT 12 - 7
PRINT 5 + 8 - 3 + 2 - 1
```

Notice that the numbers with the + and - signs are written all on one line, and the computer will add or subtract from left to right.

Of course the computer can multiply and divide also. To multiply, use the * (asterisk) symbol. To divide, use the / (slash) symbol. Remember that everything has to be on one line. To multiply 3 times 4, the equation is represented by 3 * 4. In division, the first number is divided by the second number—12/4 means 12 divided by 4.

Now the order of operations becomes important. The standard rule is to calculate from left to right, first executing the multiplication and division, and then the addition and subtraction. If we have an equation 8 + 12/4, the division 12/4 is performed first. The result, 3, is then added to 8, for a final result of 11.

Let's suppose you really wanted 8 added to 12 first, and then that sum divided by 4. On paper, you could write 8 + 12 as the numerator of a fraction; then under the bar, write the 4 to indicate division. However, with computers, the expression must be written all on one line. Parentheses may be used to group numbers. (Be sure you always have a *matching pair* of paren-

thesis.) So for our example, we can write (8 + 12)/4. This time the result is 20/4, or 5. You might want to experiment with various expressions using all the operator signs and different placements of the parentheses in PRINT statements.

One more step in the order of operations: Exponents are performed *before* the multiplication and division operations. The exponentiation operator is an up arrow (↑) on Commodore computers and a caret (^) on most others. For example, 5 + 4 ^ 3 * 10 starts with 4 ^ 3, which is 4 raised to the third power (cubed), or 64. The next operation is the multiplication 64 * 10, or 640. The next operation is the addition 5 + 640, or 645, for the final result.

Expressions With Variables

When you're comfortable using actual numbers (constants), try using numeric variables. A variable is simply a letter or name that represents a number. If you've had any algebra, you'll recall using letters in numeric expressions or formulas. For example, A=L*W is a formula for *area* equals *length* times *width*. When you solve a problem, you'll have a number for L and a number for W, which results in a number for A. Here's a sample program:

```
100 L=10
110 W=5
120 A=L * W
130 PRINT A
```

Line 120 performs the operation of multiplying the numbers represented by the variables L and W; then line 130 prints the number represented by the variable A. If you wish, you can delete line 120 and just use PRINT L * W for line 130. The result is the same.

The arithmetic symbols and order of operations are the same for variables as they are for constants.

In your program, you may

make calculations first and then use the result in another statement, or you can indicate calculations within the statement. Assuming all variables have previously been defined, these two program segments will have the same results:

```
500 X=A * B + C
510 Y=D/E
520 PRINT X,Y
```

or

```
500 PRINT A * B + C,D/E
```

So far we've seen arithmetic in the PRINT statement, but this is not the only place you can use numbers. Any statement requiring numeric parameters can use actual numbers (constants), variables, or numeric expressions. You can use these arithmetic operations to make calculations for the numbers required in other statements. For example, you may need to calculate a row and a column to begin printing or drawing. You may want to draw to one point with coordinates calculated in terms of the coordinates of another point. You may want to use SOUND commands with variables and expressions rather than constant numbers. FOR-NEXT statements may have variable index limits. IF-THEN statements may compare constants, or they may compare constants with variables, or they may compare numeric expressions. Functions such as SIN(X) can use numeric expressions in the argument. Here are some examples:

```
200 PRINT INT(A * B * RND(1) + 1)
210 IF BLUE>RED + 1 THEN 500
220 PRINT AT (ROW + 2,COL + 5),
    NAMES$
230 SOUND F,D * 4,V + 2
240 CIRCLE (X * 8 + A,Y + B),R + 5
250 LINE (X,Y)-(X + L,Y + W)
260 PRINT SGN(D * E + (F - G)/8)
270 FOR K=A + B TO C - D STEP
    (-1) * 5
```

©



Desktop Video

Because IBM PCs and compatibles are so firmly entrenched in the business microcomputer market, any new, non-compatible computer has got to offer something that IBM doesn't. For the Macintosh, desktop publishing has become a Trojan Horse, an application that's allowed it to breach the corporate walls. For the Amiga, desktop video may be the application that secures its niche in the marketplace.

The Turnaround

The Amiga's design makes it particularly well-suited for video applications, since its own video output more closely follows broadcast television standards than other personal computers. Since it was anticipated right from the start that the Amiga would be used to combine computer graphics with live video, provisions were made in its hardware and software design to support such applications. The Amiga is the only personal computer you can buy that has video input lines built right into the video output port. Commodore has been a bit slow to bring to market the hardware peripherals necessary to realize the Amiga's video potential, but things seem to be turning around at last.

One of the most exciting developments has been Commodore's recent release of the Genlock interface. Genlock is widely misunderstood among Amiga owners, because it's more of a video peripheral than a computer add-on. In effect, it turns the Amiga into a video processing accessory for your VCR, by combining a live-action video signal from a video camera or VCR with the Amiga's great computer graphics. The resulting picture can be recorded by another VCR.

Genlock replaces everything on the screen that's colored in the background color (that takes its color

from hardware color register 0). On the Workbench screen, for example, every dot that normally appears as blue is replaced by the live video picture. Software doesn't have to do a thing to work with Genlock. As soon as you attach the interface and power up the computer, the operating system sees it's there and allows you to transparently replace the background computer video color with live action video.

If Genlock only offered simple video titling, it would still allow the Amiga to surpass dedicated units costing a lot more. But Genlock lets the Amiga do things like interpose animated "characters" in the middle of live action video. Although the Genlock's signal quality isn't of the broadcast quality needed by television stations, it's certainly good enough for industrial use.

Amiga Live!

If Genlock converts computer graphics into video graphics, a video digitizer does almost exactly the opposite. It takes video graphics and changes them into computer graphics. Commodore has announced a product called Amiga Live!, which will let you freeze a single frame (or series of frames) from live action color video, and then display it on the Amiga in its own computer-graphics format. The resulting picture could be saved to disk, and later loaded into paint programs where they could be edited further.

Commodore-Amiga hasn't delivered on its promised "framegrabber" as of yet. The Digi-View Digitizer from NewTek, however, is an excellent low-cost alternative. Unlike Amiga Live!, Digi-View works too slowly to capture individual frames from live action video. In order to produce a color picture, it must capture three separate "views" of the images, repre-

senting the red, green, and blue components of the picture. To screen out other colors, a plastic wheel containing red, green, and blue filters is rotated in front of the camera lens. The whole process may take half a minute, but the quality of the pictures produced is simply astounding. Also, the Digi-View software is extremely versatile, allowing you to capture pictures in virtually any of the Amiga's video modes, including the Hold and Modify mode in which up to 4096 colors can be displayed on screen at once. When capturing pictures in the normal 32-color mode, you can specify the number of colors (1-32) to be used, and even specify the actual color palette. In this way, you can conform the image to the color palette used by an existing paint program image.

One drawback of the Digi-View system is that you normally can't use it to digitize a still frame from a VCR in color, since you can't apply the colored filters to the image. With the Genlock interface, however, you can run the video into the Genlock (set it to video only), and connect the separate red, green, and blue outputs from the Genlock's 23-pin output connector in turn to the Digi-View. The resulting Digi-View picture is quite good, as long as your VCR provides a good still-frame picture.

The applications for the Genlock and video digitizers already exist. Genlock can be used for the kind of video production work that goes on daily in every cable TV company, college video lab, and industrial video shop. It's just that up until now, it took thousands of dollars to put together the equipment needed to perform such tasks. With the Amiga, most of what's required is already sitting on your desktop. ©



A Bit Of BASIC

I'm never completely pleased with the output from off-the-shelf programs—even the good ones. There are many commands I'd like to add to word processing programs, and the screens in my accounting software—one of the most popular programs around—could be redesigned to eliminate clutter. I'd love to make these changes, but without source code and, in many cases, a knowledge of the native language, it's just not possible. Whenever I can do something about the design of the output, though, I usually spend—or waste—a lot of time rearranging things. Thus the focus of this month's column: in particular, *The Personal Ancestral File*, a program from The Church of Jesus Christ of Latter-day Saints; in general, how to use binary data.

Last October, I told you what a terrific program *PAF* is for recording genealogy and family records. And it still is. It's just that there are a few things I'd like to change on the printouts. I want both birth and death dates printed, and the age at death calculated and printed in parentheses where appropriate; I'd like ID numbers not printed; and I want a different page header with certain lines printed in bold. Minor changes, really, if only I could get my hands on the source code, which I've been told is written in C.

I can't change *PAF*, but I can write my own programs to organize and print data from *PAF* files. Basically the program uses four files, and I know what their formats are—which at first appears to be a problem for BASIC.

The main file contains one record indicating each individual's date of birth, date of death, and sex, as well as a pointer to the name, which is recorded in another file. The three things I want from this file—the information I'll use in my own program—are the name point-

er and the two dates. The problem is that the pointer and dates are not saved in the file as readable text, such as January 31, 1960 or 1-31-60. To conserve space (and make things difficult for people like me), a date is represented in a three-byte binary field. That is, it's compressed to occupy the same space as the three characters *ABC*.

Information In Bits

Here's how it's done—remember, one byte has 8 bits, so the date occupies 24 bits. The number for the year is contained in the first byte and half of the second byte—a total of 12 bits; the next 5 bits represent the month; the next 5 are the day; and the last 2 are a status code which we don't need. If you've been keeping count, you'll know that the last bit in the month is also the first bit in the third byte. Now how can you work with data that's partly in one character and partly in another? Actually, this is a common way to represent numbers and other data, and many programming languages like PL/1, C, and Pascal have ways to work directly with binary data. BASIC doesn't.

But BASIC has something called logical operators that work at the bit level. Let's see how to grab the day number (in the third byte). First we read a record and assign the three date bytes to X\$, Y\$, and Z\$, respectively. Now we want to convert a part of Z\$ to a number. The first bit we ignore because it's really the last bit in the month, and the last two we ignore since we don't need them.

The logical AND operator can be used to "wipe out" unwanted bits. If we AND the Z\$ variable with the bit pattern 0111100 we'll be erasing bits 7, 1, and 0 (bits are numbered from right to left); the bits for the day—6, 5, 4, 3, and 2—are unchanged by ANDing with the

1's in the bit pattern. So our BASIC statement might read DAY = (ASC(Z\$) AND 0111100), except BASIC doesn't have a binary data type and won't understand the 0's and 1's. We have to convert the binary pattern to hexadecimal—a base 16 numbering system—in order to write it in BASIC. To do this, divide the eight bits into two groups of four and then convert to the appropriate hex value. (Many programming books show how to do this.) The bit pattern 0111 1100 is the same thing as hex 7C, which is written in BASIC as &H7C. So our statement should be DAY = (ASC(Z\$) AND &H7C). Note that we use the ASC function to convert the character value of Z\$, which is what we had to use to read data from the file, to a numeric value.

If you print the variable DAY, you still won't get a number between 1 and 31. We aren't quite through. Here's a decimal analogy: Suppose we had the number 432651 and wanted to make a new number out of the middle two digits. After the ANDing process we would have 002600, which is 2,600—and we are after the 26 part. If we were working in decimal, we'd simply divide the ANDed value by 100 (10²) to lose the rightmost two places and come up with 26. Since we're working in binary, we divide by 4 (2²) to shift two places to the right. Thus DAY = (ASC(Z\$) AND &H7C) / 4 is the correct BASIC statement; printing DAY will print a value between 1 and 31.

To try it yourself, assign Z the initial value of 234. Since we're not reading data from a file and you would have a difficult time typing the character represented by the ASCII value 234, we won't need a character variable or the ASC function. Just make Z equal 234 and decode the day contained therein: Thanksgiving Day, 1987. ©

\$15,000.00 Programming Contest!

COMPUTE!'s PC Magazine
For IBM PCs & Compatibles

First Prize \$7,500.00

Second Prize \$2,500.00

Five Honorable Mentions \$1,000.00 each

COMPUTE! Publications, Inc., a longtime leader in personal computer publishing, is launching a new magazine this summer: COMPUTE!'s PC Magazine for IBM PCs & Compatibles. Each bimonthly issue will include a floppy disk filled with programs, source code, and other useful information. We're looking for the very best original software for IBM PCs, XTs, and compatibles, and are sponsoring a programming contest with a total of \$15,000.00 in prize money for the top six winners. That's \$7,500.00 for First Prize; \$2,500.00 for Second Prize; and five Honorable Mentions at \$1,000.00 each. In addition, the winners will receive our standard purchase fees for publication in our magazine and royalties if republished in COMPUTE! books.

Even if your contest entry doesn't win a prize, you will still earn purchase fees if we accept your program for publication.

Interested? If so, read these rules:

1. Entries must be your original work, previously unpublished in any form. All those whose programs are accepted will be required to affirm this in writing.

2. You can submit as many entries as you want, but we cannot consider programs which have been either entered in other contests or submitted for publication elsewhere at the same time.

3. The contest deadline is October 31, 1987. All entries must be received at our offices by this date. Programs submitted after this date will still be considered for publication, but will not be entered in the contest. If we purchase an entry for publication before the deadline, the entry is still eligible to win.

4. Entries are allowed (and encouraged) in virtually all software categories: home and business applications, education, recreation, telecommunications, graphics, sound and music, and utilities.

5. Entries may be written in any programming language—including BASIC, C, machine language, Pascal, and Modula-2—as long as they meet two requirements. First, if you're using a compiled language, the compiled object or runtime code must be a self-standing program that can be run by someone who doesn't own a copy of the language. (Interpreted BASIC is an exception. It can be assumed that nearly everyone owns a copy of BASICA or GW-BASIC.) Second, we must be able to legally distribute the program without incurring licensing fees or other obligations to the maker of the language. If you're not sure whether a certain language qualifies, contact its maker for clarification.

6. Entries must be submitted on 5¼-inch floppy disks. If your program is written in a compiled language, you must submit both the runtime code and all of the source code required to compile the program.

7. Entries must be accompanied by an article which explains how to use the program and what it does. If your program employs any new or unusual techniques that you think will be of interest to other programmers, you can also describe how the program works. (If you feel that writing is not your strong point, please do not hesitate to enter; this is a programming contest and the entries will be judged solely on the basis of the programs submitted.)

8. Submissions which do not win a prize and are not accepted for publication will be returned only if accompanied by a self-addressed, stamped mailer.

9. The staff of COMPUTE! Publications, Inc., will judge the contest, and all decisions regarding contest entries and acceptances will be solely at the discretion of COMPUTE! Publications, Inc. All decisions are final. This includes decisions regarding creativity, similarity among entries, and general suitability.

10. Winners will be announced by COMPUTE! Publications, Inc., in early 1988.

11. This contest is void where prohibited by law. Full-time, part-time, and previous employees of COMPUTE! Publications, Inc. and Capital Cities/American Broadcasting Corporation are ineligible for the contest, but may still submit work for publication at standard rates.

Every contest entry must include this signed form:

I warrant that the program presently entitled _____ is my own original work and that the work has not been submitted for consideration elsewhere, nor has it been previously published in any form. If my work is accepted by you, I understand that your decision as to the selection of winners and awarding of prizes is final and without recourse on my part. Should you select my submission, I understand that I will receive no payments until I sign your standard contract, which includes assignment of the copyright of the program to COMPUTE! Publications, Inc., and that you may use my name and image in promotional materials and other forms. (If you are under age 18, your parent or legal guardian must sign for you.)

Signature: _____

Address entries to: PC Programming Contest
COMPUTE! Publications, Inc.
P.O. Box 5406
Greensboro, NC 27403

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

64 RAMdisk

Hubert Cross

This Commodore 64 utility creates an electronic disk drive that's much faster than an ordinary disk drive. Since it uses "hidden" memory, the RAMdisk doesn't reduce the amount of programming space available for your use. No machine language knowledge is needed to use the program.

A RAM disk is a familiar device to many personal computer owners. In simple terms, a RAM disk emulates a disk drive entirely in the computer's RAM (Random Access Memory), allowing you to store and retrieve files much faster than you can from a mechanical disk drive. The Amiga, for instance, includes a built-in RAM disk as part of its system software; and RAM disks are popular utility programs for computers such as the Atari ST and IBM PC/PCjr.

"64 RAMdisk" is a RAM disk for the Commodore 64 which doesn't subtract a single byte from the space normally available for programming. You control the RAMdisk with simple BASIC commands, and the program is compatible with the "DOS Wedge" utility (supplied on the 1541/1571 Test/Demo disk), "TurboTape" (COMPUTE!, January 1985), and "TurboDisk" (COMPUTE!, April 1985).

Typing In 64 RAMdisk

Because 64 RAMdisk is written in machine language, the program must be typed with the "MLX" machine language entry program printed elsewhere in this issue. Here are the addresses you need to enter 64 RAMdisk with MLX:

Starting address: 0801
Ending address: 12A8

64 RAMdisk is designed to load and run exactly like a BASIC program. Load the program with LOAD "RAMDISK",8 for disk or LOAD "RAMDISK" (or simply LOAD) for tape. After the program loads, type RUN and press RETURN. The program relocates its code to the safe memory area beginning at location 49152 and prints the message RAM DISK ACTIVATED. 64 RAMdisk is now ready to use.

RAMdisk Commands

Following is a list of 64 RAMdisk commands. All of these commands work in BASIC direct mode (when you aren't running a program).

DIR. This command displays a directory of the files in the RAMdisk. For instance, if you type DIR and press RETURN, 64 RAMdisk prints a directory of the RAMdisk. The number of bytes free for use is printed at the bottom of the directory display.

NAME. The NAME command can be used to change the RAMdisk's name, which appears in reversed characters at the top of the directory display.

RSAVE. The RSAVE command saves a file to the RAMdisk, storing a copy of the BASIC program currently in memory. Here is the correct syntax for the command:

RSAVE "PROGRAM"

Of course, you should substitute the name of your program for PROGRAM in this example. If you forget to specify a filename, 64 RAMdisk prints the error message MISSING FILENAME and doesn't save anything. The filename can be any combination of 15 or fewer characters.

RLOAD. The RLOAD command copies any program in the RAMdisk back to BASIC. For example, the statement RLOAD "EXAMPLE" loads the program named EXAMPLE from the RAMdisk, storing a copy of it in the usual BASIC program space.

SCRATCH. The SCRATCH command deletes a file from the RAMdisk. For instance, the statement SCRATCH "PROGRAM" removes the file named PROGRAM from the RAMdisk. If you attempt to scratch a file that doesn't exist in the RAMdisk, the program prints the error message FILE NOT FOUND and doesn't scratch anything.

RENAME. This command changes the name of a file in the RAMdisk. Here is the syntax to use:

RENAME "OLDNAME","NEWNAME"

The RENAME command requires two filenames: the name of the existing file (OLDNAME in this example) and the new name which you want that file to have (NEWNAME in this example). The filenames are separated with a comma. The error message FILE NOT FOUND appears if you attempt to rename a nonexistent file. The error message FILE EXISTS appears if you try to use a new name which already exists in the RAMdisk. (Every file in the RAMdisk must have a unique name.)

REPLACE. This command replaces the designated RAMdisk file with the BASIC program in memory, using the same filename. For instance, REPLACE "TEST" deletes the program TEST from the RAMdisk and saves the BASIC program in memory, using the filename TEST. This is normally done when

you have made changes to a program and wish to save the revised version with the same name.

RNEW. The RNEW command does a NEW of the RAMdisk, erasing every file that it contains. *Be very careful when using this command*, since the program does not ask you to confirm this action, and there is no easy way to recover files after an RNEW. If you include a name after RNEW, this command renames the RAMdisk with the name specified. For instance, this command erases everything in the RAMdisk and renames it as MYDISK:

```
RNEW "MYDISK"
```

EXIT. This command disables the RAMdisk and gives you the option to save the entire contents of the RAMdisk in a single tape or disk file. When you type EXIT and press RETURN, the program prints this prompt:

```
EXIT RAM DISK  
ARE YOU SURE? (Y/N)
```

If you type N at this prompt, nothing happens. If you type Y at this prompt, the program copies the 64 RAMdisk machine language program, as well as all the files in the RAMdisk, back into the BASIC program space. This is done so that you can save the entire RAMdisk as a single file, using a normal SAVE command from BASIC. The advantage of this method is that you can reload the RAMdisk program *and all of the saved files* in one operation, at the beginning of your next programming session. You do not need to use any special tricks to save this file. Here is a typical SAVE command:

```
SAVE "CABOODLE",8
```

This example saves 64 RAMdisk and the entire contents of the RAMdisk under the filename CABOODLE. If you replace the 8 with a 1, the file is saved to tape instead of disk. (The resulting file will be considerably longer than 64 RAMdisk itself, since it contains a copy of every file in the RAMdisk, as well as a copy of 64 RAMdisk.)

Once you have saved a master file, you can reload 64 RAMdisk and the individual files with the same load and run commands you would ordinarily use to activate the RAMdisk. In this case, for instance,

you would use these commands:

```
LOAD "CABOODLE",8  
RUN
```

If you type RUN and press RETURN at this point, the program moves the 64 RAMdisk code to its normal location beginning at location 49152, then transfers each of the saved files to the RAMdisk, too. When the startup message appears, the RAMdisk is ready to use and all of the files are in place.

Notice that the filename used with EXIT has no connection with the name of the RAMdisk itself (see NAME) or the names of individual files contained in the RAMdisk. As a practical matter, however, you will probably want to use a name that reminds you what individual files the master file contains. If you previously used NAME to give the RAMdisk a meaningful name, you can use the same name when saving the entire disk with EXIT.

RAMdisk Notes

64 RAMdisk is designed as a convenience for saving and loading BASIC programs, not as a total replacement for a disk or tape drive. Thus, it supports only one type of file—a BASIC program (PRG) file—and only one form of file access (saving and loading). You cannot use the RAMdisk for other types of files, such as sequential (SEQ) files. And, for instance, you cannot OPEN a file in the RAMdisk for reading or writing, even though those are legitimate operations for program (PRG) files on tape or floppy disk.

Like all RAMdisks, 64 RAMdisk is volatile, meaning that it disappears completely, together with all its contents, when you turn off the computer. For this reason, you should make frequent backup copies of RAMdisk programs on disk or tape. To make a backup copy, load the program into BASIC memory with RLOAD; then save it to disk or tape in the usual way.

This program occupies the memory area beginning at location 49152 (\$C000), so you cannot use it with any other machine language program or utility that occupies the same space. Because this program uses the "hidden" RAM underlying the 64's ROM (Read-Only Memory) chips, it is also incompatible

with programs which use that area of RAM. 64 RAMdisk *does* work with either TurboTape or TurboDisk, but not with both at the same time, since those programs are incompatible with one another. You must relocate TurboDisk, as explained in the TurboDisk article, before using it with 64 RAMdisk; the best place to put TurboDisk is as close as possible to the top of BASIC RAM.

The 64 has a total of 16K (16,384 bytes) of RAM under its ROM chips. Half of this lies under the BASIC ROM, and the other half lies under the Kernal operating system ROM. Not all of this RAM can be used for file storage with 64 RAMdisk. Every program stored in the RAMdisk requires an extra 18 bytes for a directory entry, 15 bytes for a filename entry, 1 byte as a filename marker, and 2 bytes for a pointer to the beginning of the next program.

Programs are stored beginning at the bottom of the RAM under BASIC and growing upward, toward higher memory locations. The directory begins at the top of the RAM under the Kernal and grows downward, toward lower memory locations. If you fill the 8K space under BASIC with programs, 64 RAMdisk uses as much of the RAM under the Kernal as needed. If you try to save a program that's bigger than the amount of free space left in the RAMdisk, the program prints the error message RAM DISK FULL and doesn't save anything.

It's theoretically possible to create so many individual files that the RAMdisk directory would fill all of the RAM under the Kernal ROM. However, since it would require more than 454 files to overflow the directory, 64 RAMdisk does not check for this unlikely situation, and does not print an error message if it occurs.

64 RAMdisk

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

```
0801:0B 08 00 00 9E 32 30 36 EC  
0809:31 00 00 00 4C FD 0F 4C 76  
0811:1D C2 4C 30 C2 4C 5F C5 B8  
0819:4C 10 C2 4C 5B C2 4C 47 36  
0821:C3 4C 07 C4 4C 3E C3 4C 82  
0829:55 C5 4C 1C C9 20 55 C0 DA  
0831:A9 A9 8D 04 03 A9 C0 8D 40  
0839:05 03 A9 69 8D 06 03 A9 8C  
0841:C1 8D 07 03 A9 D0 8D 08 5A
```

0849:03 A9 C1 8D 09 03 A9 94 92
0851:A0 C0 4C 1E AB A9 00 8D DE
0859:FE FF 8D 78 CA A9 80 8D AD
0861:FF FF 8D 79 CA A9 DC 85 F6
0869:FB 8D 7A CA A9 FF 85 FC 2C
0871:8D 7B CA 60 0F B9 84 C0 AF
0879:99 EE FF 88 10 F7 A9 60 AE
0881:8D 73 C0 60 32 34 4B 20 6C
0889:52 41 4D 20 44 49 53 4B F7
0891:00 00 00 00 0D 52 41 4D 23
0899:20 44 49 53 4B 20 41 43 C9
08A1:54 49 56 41 54 45 44 0D 5A
08A9:00 A6 7A A0 04 84 0F BD CA
08B1:00 02 10 07 C9 FF F0 3E 23
08B9:EB D0 F4 C9 20 F0 37 85 66
08C1:08 C9 22 F0 56 24 0F 70 6D
08C9:2D C9 3F D0 04 A9 99 D0 A2
08D1:25 C9 30 90 04 C9 3C 90 46
08D9:1D 84 71 A0 00 84 0B 88 82
08E1:86 7A CA C8 E8 BD 00 02 F9
08E9:38 F9 9E A0 F0 F5 C9 80 E5
08F1:D0 30 05 B0 A4 71 E8 C8 4D
08F9:99 FB 01 B9 FB 01 F0 38 8F
0901:38 E9 3A F0 04 C9 49 D0 AA
0909:02 85 0F 38 E9 55 D0 9F C8
0911:85 08 BD 00 02 F0 DF C5 F8
0919:08 F0 DB C8 99 FB 01 E8 1B
0921:D0 F0 A6 7A E6 0B C8 B9 03
0929:BD A0 10 FA B9 9E A0 D0 3E
0931:B4 F0 0F BD 00 02 10 B0 7C
0939:99 FD 01 C6 7B A9 FF 85 2C
0941:7A 60 A0 FF CA C8 E8 BD C5
0949:00 02 38 F9 9E C1 F0 F5 56
0951:C9 80 D0 04 05 0B D0 9C 55
0959:A6 7A E6 B0 C8 B9 9D C1 15
0961:10 FA B9 9E C1 D0 E0 F0 5F
0969:CA 30 03 4C F3 A6 C9 FF DF
0971:F0 F9 24 0F 30 F5 C9 CB A9
0979:B0 03 4C 24 A7 38 E9 CC 2E
0981:AA 84 49 A0 FF CA F0 08 52
0989:C8 B9 9E C1 10 FA 30 F5 21
0991:C8 B9 9E C1 30 05 20 47 83
0999:AB D0 F5 4C EF A6 4E 41 31
09A1:4D C5 52 45 4E 41 4D C5 42
09A9:53 43 52 41 54 43 C8 52 28
09B1:4E 45 27 52 53 41 56 C5 6E
09B9:52 4C 4F 41 C4 44 49 D2 A2
09C1:52 52 55 CE 52 45 50 4C BD
09C9:41 43 C5 45 58 49 D4 00 EB
09D1:20 73 00 20 D9 C1 4C AE EF
09D9:A7 C9 CC 90 14 C9 D6 B0 FA
09E1:10 38 E9 C0 0A A8 B9 78 73
09E9:C1 48 B9 F7 C1 48 4C F3 E0
09F1:00 20 79 00 4C ED A7 0E B2
09F9:C0 11 C0 14 C0 17 C0 1A 08
0A01:C0 1D C0 20 C0 23 C0 26 11
0A09:C0 29 C0 2C C0 2F C0 60 47
0A11:20 79 00 D0 03 4C 55 C0 55
0A19:20 1D C2 30 F8 20 81 C7 F2
0A21:A8 A9 00 99 EE FF 88 B1 C7
0A29:22 99 EE FF 88 10 F8 60 69
0A31:20 81 C7 20 2D C7 90 03 5D
0A39:4C 9E C7 20 FD AE 20 81 82
0A41:C7 48 A6 FB 86 6B A6 FC 0C
0A49:86 6C 20 2D C7 90 1D A5 F2
0A51:6B 85 FB A5 6C 85 FC 68 32
0A59:4C 90 C7 20 79 00 D0 03 23
0A61:4C A1 C7 20 81 C7 48 20 DA
0A69:2D C7 B0 03 4C AD C7 A5 9A
0A71:2D 38 E5 2B 85 FD 85 6D 36
0A79:A5 2E E5 2C 85 FE 85 6E 0D
0A81:AD 7A CA 38 ED 78 CA 85 54
0A89:6B AD 7B CA ED 79 CA 85 4B
0A91:6C AD 78 CA C9 00 AD 79 26
0A99:CA E9 C0 B0 0D A5 6B 38 BE
0AA1:E9 00 85 0B A5 6C E9 20 E4
0AA9:85 6C A5 6B 38 E9 12 85 1A
0AB1:6B A5 6C E9 00 85 6C 90 90
0AB9:0A A5 6B C5 FD A5 6C E5 4B
0AC1:FE B0 03 4C B4 C7 A5 2B E1
0AC9:8D DD C2 A5 2C 8D DE C2 E6
0AD1:AD 78 CA 8D E0 C2 AD 79 F3
0AD9:CA 8D E1 C2 AD FF 8D 1A

0AE1:FF FF EE DD C2 D0 03 EE FF
0AE9:DE C2 EE E0 C2 D0 03 EE 58
0AF1:E1 C2 AD E0 C2 C9 00 D0 79
0AF9:11 AD E1 C2 C9 C0 D0 0A 67
0B01:A9 00 8D E0 C2 A9 E0 8D B7
0B09:EL C2 A5 6D D0 02 C6 6E D6
0B11:C6 6D D0 C8 A6 6E D0 C4 E1
0B19:A0 10 AD E0 C2 91 FB 8D 29
0B21:78 CA C8 AD E1 C2 91 FB 53
0B29:8D 79 CA 68 20 90 C7 AD C4
0B31:7A CA 38 E9 12 8D 7A CA 63
0B39:B0 03 CE 7B CA 60 20 51 63
0B41:C3 20 90 FF 4C 59 A6 24 8C
0B49:9D 10 F3 20 51 C3 4C 6C 51
0B51:CA 20 79 00 D0 03 4C A1 D0
0B59:C7 20 81 C7 20 2D C7 90 DD
0B61:03 4C 9E C7 A0 22 78 A5 80
0B69:01 48 A9 35 85 01 B1 FB 2A
0B71:85 6D C8 B1 FB 85 6E A0 4D
0B79:10 B1 FB 85 2D C8 B1 FB C7
0B81:85 2E 68 85 01 58 A5 2D 2D
0B89:38 E5 6D 85 FD A5 2E E5 04
0B91:6E 85 FE A5 2D C9 00 A5 B0
0B99:2E E9 C0 90 17 A5 6D C9 56
0BA1:00 A5 6E E9 E0 B0 D0 A5 17
0BA9:FD 38 E9 00 85 FD A5 FE 78
0BB1:E9 20 85 FE A5 2B 85 FB 46
0BB9:18 65 FD 85 2D A5 2C 85 2B
0BC1:FC 65 FE 85 2E A0 00 78 54
0BC9:A5 01 48 A9 35 85 01 B1 0A
0BD1:6D 91 FB 68 85 01 58 E6 D0
0BD9:6D D0 02 6E 6E A5 6D C9 38
0BE1:00 D0 0E A5 6E C9 C0 D0 35
0BE9:08 A9 00 85 6D A9 E0 85 20
0BF1:6E E6 FB D0 02 E6 FC A5 D0
0BF9:F7 38 E9 01 85 FD B0 02 F1
0C01:C6 FE 05 FE D0 C1 60 A9 C4
0C09:0D 20 D2 FF A9 30 20 D2 2B
0C11:FF A9 20 20 D2 FF A9 12 95
0C19:20 D2 FF A9 22 D2 FF C7
0C21:78 A5 01 48 A9 35 85 01 B1
0C29:AD FE FF 85 6D AD FF F7 52
0C31:85 6E A9 EE 85 FB A9 FF 3B
0C39:85 FC A0 00 B1 FB 99 EC 05
0C41:C7 F0 03 C8 A0 F6 68 85 1F
0C49:01 58 A9 EC A0 C7 20 1E 7E
0C51:AB A9 22 20 D2 FF A9 0D E6
0C59:20 D2 FF 20 E1 FF F0 A6 CF
0C61:A5 FB 38 E9 12 85 FB B0 40
0C69:02 C6 FC A5 FB CD 7A CA 05
0C71:D0 0A A5 FC AD 7B CA D0 BB
0C79:03 4C 09 C5 A0 10 78 A5 7F
0C81:01 48 A9 35 85 01 B1 FB 44
0C89:85 69 C8 B1 FB 85 6A 68 26
0C91:85 01 58 A5 69 38 E5 6D 77
0C99:85 FD A5 6A E5 6E 85 FE 42
0CA1:A5 69 38 E9 00 A5 6A E9 E1
0CA9:C0 90 18 A5 6D 38 E9 00 C3
0CB1:A5 6E E9 E0 B0 0D A5 FD 86
0CB9:38 E9 00 85 FD A5 FE E9 2F
0CC1:20 85 FE A5 69 48 A5 6A A7
0CC9:48 A6 FD A5 FE 20 CD BD 9B
0CD1:68 85 6E 68 85 6D A9 05 0E
0CD9:85 D3 20 6C E5 A9 22 20 AE
0CE1:D2 FF 78 A5 01 48 A9 35 7E
0CE9:85 01 A0 00 B1 FB 99 EC B6
0CF1:C7 F0 03 C8 D0 F6 68 85 CF
0CF9:01 58 A9 EC A0 C7 20 1E 2F
0D01:AB A9 22 20 D2 FF 4C 56 27
0D09:C4 AD 7A CA 38 ED 78 CA 22
0D11:85 6D AD 7B CA ED 79 CA 82
0D19:85 6E AD 78 CA C9 00 AD F9
0D21:79 CA E9 C0 B0 0D A5 6D 66
0D29:38 E9 00 85 6D A5 6E E9 FA
0D31:20 85 6E A6 6D A5 6E 20 F3
0D39:CD BD A9 06 85 D3 20 6C 67
0D41:E5 A9 48 A0 C5 4C 1E AB 13
0D49:42 59 54 45 53 20 46 52 B3
0D51:45 45 2E 0D 00 20 81 C7 41
0D59:48 20 62 C5 4C 6F C2 20 0E
0D61:81 C7 20 2D C7 90 03 4C D7
0D69:9E C7 A0 22 78 A5 01 48 9F
0D71:A9 35 85 01 B1 FB 85 66 5D

0D79:CB B1 FB 85 67 A0 B1 CB
0D81:FB 85 69 C8 B1 FB 85 6A A7
0D89:68 85 01 58 A5 69 38 E5 08
0D91:66 85 FD A5 6A E5 67 85 99
0D99:FE A5 69 C9 00 A5 6A E9 7B
0DA1:C0 90 17 A5 66 C9 00 A5 7D
0DA9:67 E9 E0 B0 0D A5 FD 38 4C
0DB1:E9 00 85 FD A5 FE E9 20 6E
0DB9:85 FE AD 7A CA 18 69 12 4F
0DC1:C5 FB D0 28 AD 7A CA 18 5F
0DC9:69 12 AD 7B CA 69 00 C5 4C
0DD1:FC D0 19 A5 66 8D 78 CA 41
0DD9:A5 67 8D 79 CA AD 7A CA B6
0DE1:18 69 12 8D 7A CA 90 03 A0
0DE9:EE 7B CA 60 A0 00 78 A5 55
0DF1:01 48 A9 35 85 01 B1 69 24
0DF9:91 66 68 85 01 58 E6 69 7C
0E01:D0 02 E6 6A A5 69 C9 00 6F
0E09:D0 0E A5 6A C9 C0 D0 08 E7
0E11:A9 00 85 69 A9 E0 85 6A 8F
0E19:E6 66 D0 02 E6 67 A5 66 03
0E21:C9 00 D0 0E A5 67 C9 C0 3C
0E29:D0 08 A9 00 85 66 A9 E0 DE
0E31:85 67 A5 69 CD 78 CA D0 EB
0E39:B5 A5 6A CD 79 CA D0 AE 0B
0E41:AD 78 CA 38 E5 FD 85 6B CC
0E49:AD 79 CA E5 FE 85 6C AD E6
0E51:78 CA C9 00 AD 79 CA E9 68
0E59:C0 90 17 A5 6B C9 00 A5 5F
0E61:6C E9 E0 B0 0D A5 6B 38 63
0E69:E9 00 85 6B A5 6C E9 20 B4
0E71:85 6C A5 6B 8D 78 CA A5 60
0E79:6C 8D 79 CA A5 6B 38 E9 82
0E81:01 85 6D A5 FC E9 00 85 9C
0E89:6E A0 00 78 A5 01 48 A9 F7
0E91:35 85 01 B1 6D A0 12 91 88
0E99:6D 68 85 01 58 A6 6D D0 50
0EA1:02 C6 6E C6 6D A5 6D CD 55
0EA9:7A CA D0 DD A5 6E CD 7B AB
0EB1:CA D0 D6 AD 7A CA 18 69 B5
0EB9:12 8D 7A CA 90 03 EE 7B 28
0EC1:CA 78 A5 01 48 A9 35 85 FE
0EC9:01 A0 10 B1 FB 85 6B C8 41
0ED1:B1 FB 85 6C 68 85 01 58 F0
0ED9:A5 6B C9 00 A5 6C E9 C0 50
0EE1:08 A5 6B 38 E5 FD 85 6B F9
0EE9:A5 6C E5 FE 85 6C 28 90 5F
0EF1:17 A5 6B C9 00 A5 6C E9 66
0EF9:0E B0 0D A5 6B 38 E9 00 BE
0F01:85 6B A5 6C E9 20 85 6C 7F
0F09:A0 10 A5 6B 91 FB C8 A5 9A
0F11:6C 91 FB A5 6B 38 E9 12 4A
0F19:85 FB B0 02 C6 FC A5 FB 0A
0F21:CD 7A CA D0 9C A5 FC CD 6E
0F29:7B CA D0 95 60 85 6F A9 CC
0F31:CD 85 FB A9 6F 85 FC A5 EE
0F39:FB CD 7A CA D0 07 A5 FC AF
0F41:CD 7B CA F0 3B A0 00 78 62
0F49:A5 01 48 A9 35 85 01 B1 91
0F51:FB AA 68 85 01 58 8A D1 CD
0F59:22 D0 18 C8 C4 6F D0 E7 B9
0F61:18 78 A5 01 48 A9 35 85 47
0F69:01 B1 FB AA 68 85 01 58 52
0F71:8A F0 0D A5 FB 38 E9 12 B3
0F79:85 FB B0 BB C6 FC D0 B7 AE
0F81:60 20 9E AD 20 A3 B6 C9 4D
0F89:00 F0 16 C9 10 B0 17 60 15
0F91:A8 A9 00 91 FB 88 B1 22 0F
0F99:91 FB 88 10 F9 60 A2 02 2C
0FA1:2C A2 08 4C 37 A4 A2 C2 98
0FA9:A0 C7 4C BB C7 A2 D4 A0 62
0FB1:C7 4C BB C7 A2 DF A0 C7 58
0FB9:4C BB C7 86 22 84 23 4C 04
0FC1:47 A4 46 49 4C 45 20 4E 10
0FC9:41 4D 45 20 54 4F 4F 20 25
0FD1:4C 4F 4E C7 46 49 4C 45 65
0FD9:20 45 58 49 53 54 D3 52 DE
0FE1:41 4D 20 44 49 53 4B 20 8A
0FE9:46 55 4C CC 00 00 FF FF D0
0FF1:00 00 FF FF 00 00 FF FF 10
0FF9:00 00 FF FF A5 33 C9 01 A6
1001:A5 34 E9 80 B0 A0 A5 37 76
1009:C9 01 A5 38 E9 80 00 C0 05

```

1011:A9 00 85 33 85 37 A9 80 C6
1019:85 34 85 38 A9 00 85 61 F6
1021:A9 C0 85 62 A9 7C 38 E5 B2
1029:61 85 63 A9 CA E5 62 85 9A
1031:64 A5 63 18 69 01 85 65 9A
1039:A5 64 69 08 85 66 A5 65 69
1041:18 69 04 85 69 A5 66 69 B8
1049:00 85 6A A0 00 B1 65 8D 41
1051:78 CA 38 E9 00 85 6B C8 BB
1059:B1 65 8D 79 CA E9 80 85 79
1061:6C AD 78 CA C9 00 AD 79 02
1069:CA E9 E0 90 0D A5 6B 38 9C
1071:E9 00 85 6B A5 6C E9 20 C0
1079:85 6C A9 01 85 02 A9 08 4C
1081:85 03 A0 00 B1 02 91 61 53
1089:E6 02 D0 02 E6 03 E6 61 4A
1091:D0 02 E6 62 A6 63 D0 02 04
1099:C6 64 C6 63 A5 63 05 64 6E
10A1:D0 E2 A0 02 B1 65 8D 7A CF
10A9:CA C8 B1 65 8D 7B CA A9 87
10B1:00 85 02 A9 80 85 03 A5 D3
10B9:6B 05 6C F0 34 A0 00 B1 43
10C1:69 91 02 E6 69 D0 02 E6 23
10C9:6A E6 02 D0 02 E6 03 A5 7D
10D1:02 C9 00 D0 0E A5 03 C9 49
10D9:C0 D0 08 A9 00 85 02 A9 ED
10E1:E0 85 03 A6 6B D0 02 C6 08
10E9:6C C6 6B A5 6B 05 6C D0 D2
10F1:CE AD 7A CA 85 02 AD 7B EB
10F9:CA 85 03 A0 00 B1 69 91 76
1101:02 E6 69 D0 02 E6 6A E6 7F
1109:02 D0 02 E6 03 A5 02 05 C6
1111:03 D0 EA 20 30 C0 20 42 4F
1119:A6 4C 86 E3 A9 31 A0 C9 CD
1121:20 1E AB 20 E4 FF C9 59 66
1129:F0 2B C9 4E D0 F5 4C D2 76
1131:FF 0D 45 58 49 54 20 52 F2
1139:41 4D 20 44 49 53 4B 0D D2
1141:41 52 45 20 59 4F 55 20 16
1149:53 55 52 45 3F 20 28 59 2D
1151:2F 4E 29 20 00 20 D2 FF EB
1159:4C 8F C9 02 A9 C0 85 03 3D
1161:A0 00 B1 02 38 E9 00 91 25
1169:02 C8 B1 02 E9 C0 91 02 8C
1171:A0 00 B1 02 18 69 01 91 34
1179:02 C8 B1 02 69 08 91 02 B5
1181:A9 4C 8D 58 C9 A9 8F 8D 64
1189:59 C9 A9 C9 8D 5A C9 A9 AF
1191:00 85 61 A9 C0 85 62 A9 66
1199:01 85 2B 85 02 A9 08 85 A7
11A1:2C 85 03 A0 00 B1 61 91 C0
11A9:02 E6 61 D0 02 E6 62 E6 17
11B1:02 D0 02 E6 03 A5 61 C9 F2
11B9:7C D0 EA A5 62 C9 CA D0 A6
11C1:E4 AD 78 CA 91 02 C8 AD 51
11C9:79 CA 91 02 C8 AD 7A CA 6A
11D1:91 02 C8 AD 7B CA 91 02 5D
11D9:A5 02 18 69 04 85 02 90 B3
11E1:02 E6 03 A9 00 85 61 A9 3C
11E9:80 85 62 A0 00 A5 61 CD 2B
11F1:78 CA D0 07 A5 62 CD 79 59
11F9:CA F0 32 78 A5 01 48 A9 F6
1201:35 85 01 B1 61 91 02 68 1A
1209:85 01 58 E6 61 D0 02 E6 E2
1211:62 E6 02 D0 02 E6 03 A5 C4
1219:61 C9 00 D0 D0 A5 62 C9 19
1221:C0 D0 CA A9 00 85 61 A9 50
1229:E0 85 62 D0 C0 AD 7A CA F4
1231:85 61 AD 7B CA 85 62 78 87
1239:A5 01 48 A9 35 85 01 B1 87
1241:61 91 02 68 85 01 58 E6 09
1249:61 D0 02 E6 62 E6 02 D0 84
1251:02 E6 03 A5 61 05 62 D0 9F
1259:DE A5 02 85 2D A5 03 85 7A
1261:2E A0 05 B9 72 CA 99 04 F6
1269:03 88 10 F7 20 60 A6 4C CE
1271:86 E3 7C A5 1A A7 E4 A7 9C
1279:00 80 DC FF 00 80 DC FF 15
1281:00 00 FF FF 00 00 FF FF A5
1289:00 00 FF FF 00 00 FF FF AD
1291:00 00 32 34 4B 20 52 41 FF
1299:4D 20 44 49 53 4B 00 00 51
12A1:00 00 00 80 00 00 00 CD

```

Printing Special Characters

George Vogel

A word processor and printer can do much more than meets the eye. Here are three techniques that explore ways you can produce printed text with special characters for use in foreign-language sets or scientific notation. Although the example program is written for Apple II computers, the general technique can be adapted to any personal computer.

Easy, But Limited

The first, and most obvious, approach may be suitable for foreign-language text. Many printers have one or more international character sets, which can be selected by setting DIP switches as described in the printer manual. The various special characters are then available in place of certain less-essential characters such as brackets, braces, and so on. One limitation of this method is that it works only if your keyboard has keys corresponding to the characters which are replaced with international characters. The Apple II+ keyboard, for instance, lacks many such keys.

Let's look at a fairly typical example. The Star Micronics SG-10 printer, which is in many respects very similar to Epson printers, offers six international character sets: German, Italian, Danish, Swedish, Spanish, and French. Unfortunately, only the first four sets are complete. (The French and Spanish character sets require more letters with diacritical marks than are pro-

Word processors are generally limited in the kind of text they can print. While most can produce underlining, boldface, and superscripts, they are not usually designed to handle special characters such as foreign language or scientific symbols. In fact, your printer may already be capable of printing such special characters, but the process of using them from a word processor is often difficult. This article explores three possible solutions to the problem.

vided in the printer character set.) And there are many languages, of course, which can't be represented with any of these character sets. We'll see how those languages might be accommodated later in this article.

The Top 128

A second method of producing special characters involves the highest 128 characters in the ASCII character set. As you may know, the standard ASCII character set uses character codes 0-127 for the typical English alphabet, numerals, and punctuation. Character codes in the range 128-255 have no standard meaning, so they are often used for block graphics and various other special symbols. These sets vary greatly from one printer to another, but it's possible that yours has the characters you need for a particular application.

Although some computers can produce ASCII codes above 127 from the keyboard (often by pressing two keys at once), the Apple II series does not have this capability. How, then, can you use these special codes in a word processor? One method is to sacrifice a seldom-used pair of special printer commands.

To illustrate, say that you have a Star SG-10 printer and you are using the *Word Handler II* word processor on an Apple II computer. This particular word processor can be set up for many different printers because it allows you to specify the control code sequence (often called an *escape code*) which activates each of the printer's special features. Let's assume that you are willing to do without the ability to print superscript characters. As with many such features, using superscript involves two commands: one to turn superscript on and the other to turn it off.

The normal SG-10 escape codes for superscript are \$1B \$53 \$01 and \$1B \$54 \$1B \$48. The first character in each code—\$1B, or decimal 27—is usually called the *escape* character (hence the term *escape code*). This character simply serves as a signal that the printer should interpret the following characters as control characters—commands—rather than as something to print literally.

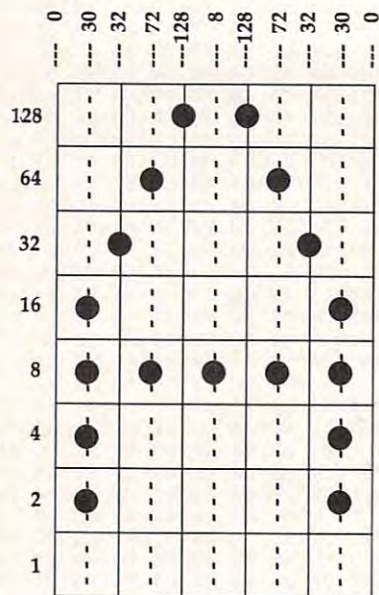
To adapt the word processor for printing the special characters, enter the printer setup feature and replace the normal escape code for *superscript on* with \$1B \$3E and the code for *superscript off* with \$1B \$3D. These are the SG-10 printer's commands for *set highest bit to 1* and *set highest bit to 0*. Since the highest bit of an eight-bit character is equal to 128, setting the highest bit to 1 has the effect of adding 128 to the ASCII code. Thus, when the first condition is activated, the printer automatically adds 128 to every character it receives, before it prints that character. This remains in effect until canceled with the second code. If you type the letter *F* (ASCII 70) in this mode, the printer prints ASCII 198 (70 + 128). On the Star printer, character 198 happens to be a degree (°) symbol, so this method offers an easy means of generating that symbol for math or scientific documents. Whenever you need to use that symbol, you must cause the word processor to issue the command it would normally use to activate superscript. You then type an *F*, followed by the command to turn off superscript. The control code goes to the printer, but the word processor treats the *F* as usual, so the screen display shows an *F* rather than the special symbol.

User-Defined Characters

A third, more versatile, approach is possible if you have a dot-matrix printer with the ability to print user-defined characters. These are simply characters whose shape you design on your own. Typically, you design the special character set in a separate operation and store it in a disk file. Before using user-defined characters, you must invoke a special command to read the new character shapes from the disk file into the computer's memory and download, or send, them to the printer.

Downloading new character definitions is possible only on a printer which contains RAM (Random Access Memory) set aside for this purpose. Under normal circumstances, the printer uses the character shapes permanently recorded in a ROM (Read Only Memory) chip. But if you store new shapes in RAM and send the appropriate command, the printer uses the RAM-

Printer Character Pattern



based shape definitions, instead.

We can use the method described in the previous section to send the command that makes the printer switch to the RAM character set. In the case of the SG-10, for instance, the code to switch to the RAM characters is \$1B \$24 \$01, and the code to switch back to the normal characters in ROM is \$1B \$20 \$00.

The details of this process will vary from one printer to the next, so consult your printer's manual for specific guidance. To give you a general idea of how it's done, we'll describe the procedure for the Star SG-10. On this particular printer, each character is formed out of dots within a matrix that's 8 dots high and 11 dots wide. The two rightmost columns are normally left blank to provide white space between characters.

On Star printers, each vertical line of a character can be described by a byte, or a number in the range 0-255. A character contains 11 vertical lines of dots. Thus, to describe the dot pattern of a character, we must supply a series of 11 numbers, each in the range 0-255.

For each number in the character definition, a dot in the lowest position is equal to 1; the next highest dot, 2; the next, 4; and so on, up to 128 in the topmost position. The figure shows how you might define

the character A for a Star SG-10 printer. The numbers on the left side of the figure indicate the value of a dot in the corresponding column. At the top of each column is the number for that vertical line of the character. This particular character can be defined with the numbers 0, 30, 32, 72, 128, 8, 128, 72, 32, 30, 0.

Three more pieces of information are needed to describe the character completely. The first two define where in the character set this character will appear. For instance, if you wanted the new character to replace the # symbol, ASCII 35, you would supply the number 35. The SG-10 requires a range of positions where the new character is to be installed, so the second value will also be 35. The third value combines information about the width of the character (for proportional spacing, if used) and information to lower the character to produce a descender, or tail. When downloading a user-defined character to this printer, you send these three values first, followed by the 11 numbers which actually define the shape of the character.

The example program follows this procedure to download a number of new characters to the printer. Lines 150-160 are necessary because the Apple printer card cannot transmit eight-bit quantities.

This method allows you to define as many new characters as you need, limited only by the number of characters available from your keyboard. In a few cases, you might want a new character set that completely replaces the normal ROM character set. It's more common to copy most of the ROM characters into RAM and redefine only part of the character set. Once the new characters have been downloaded, you can switch between them and the ROM characters at will.

You may be wondering whether these techniques will work with your particular word processor and printer. The only way to find out is to try. Printers differ very widely in their capabilities, so you will need to read your printer manual carefully before going any further. Most printer manuals include example programs in BASIC which demonstrate that printer's capabilities.

Word processors, too, handle special features in different ways. If your word processor allows you to specify the actual control codes for special features, you can implement with it any of the special features that your printer can handle. If the word processor is dedicated to specific printers and does not let you change the control codes, your only solution is to use downloaded characters, as long as your printer supports them.

Printer Example

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

```

A3 10 REM COPYRIGHT 1987 COMPUTE
! PUBLICATIONS, INC. ALL R
IGHTS RESERVED
C1 15 TEXT : HOME : PRINT "COPYR
IGHT 1987": PRINT "COMPUTE
! PUBLICATIONS, INC": PRIN
T "ALL RIGHTS RESERVED"
7A 17 FOR I = 1 TO 1000: NEXT I
28 20 D$ = CHR$ (4):E$ = CHR$ (2
7)
9C 30 DIM A(95),M$(14)
7F 40 HOME : INVERSE : PRINT "
D O W N L O A D . F R E
N C H " : NORMAL : PRI
NT
49 50 PRINT : PRINT "PRINT CHARA
CTER TABLE? (Y/N) ";; GET
PR$: PRINT : PRINT
8D 60 PRINT "(E)MPHASIZED?": PRI
NT "(P)ROPORTIONAL?": PRIN
T "(D)DOUBLE-STRIKE PROPERT
IONAL? " : PRINT : PRINT "(
OR ANY OTHER KEY FOR REGUL
AR) ";; GET PS$: PRINT
1F 70 PRINT : PRINT "DOWNLOADING
CHARACTERS - WAIT"
95 80 PRINT D$"PR#1": PRINT E$"@
";E$"*" CHR$ (0);: REM INI
T. PRINTER, COPY CHAR. ROM
TO DL RAM
18 90 ONERR GOTO 190
DB 100 :: REM : REPLACE REGULAR
BY CUSTOM CHARS IN DL RA
M
79 110 FOR K = 1 TO 14: READ M$(
K): NEXT
5D 120 S = S + 1:A(S) = M$(1)
D9 130 PRINT E$"*" CHR$ (1);
B8 140 FOR K = 1 TO 14
53 150 IF PEEK (49601) > 127 THE
N 150
C7 160 POKE 49296,M$(K)
EC 170 NEXT K
9B 180 GOTO 110
BF 190 ON PR$ = "Y" GOSUB 260: R
EM : PRINT CHAR TABLE IF
REQUESTED
98 200 :: REM : SET PRINTING ST
YLE
79 210 IF PS$ = "E" THEN PRINT E
$"E";: REM EMPHASIZED
8D 220 IF PS$ = "P" THEN PRINT E
$ CHR$ (112) CHR$ (1);: R
EM PROPORTIONAL
53 230 IF PS$ = "D" THEN PRINT E
$"G";E$ CHR$ (112) CHR$ (

```

```

1);: REM DBL-STRIKE/PROPO
RTIONAL
E8 240 PRINT : PRINT D$"PR# 0":
END
B2 250 :: REM : PRINT LIST OF D
OWNLOAD CHARS WITH THOSE
ACTIVATING THEM
34 260 PRINT : PRINT D$"PR#1": P
RINT E$"M" CHR$ (8);:CL =
32: REM L. MARGIN, CHARS
/LINE
DD 270 R = S:F = 1:L = CL
C7 280 IF R > CL GOTO 300
69 290 L = F + R - 1
2F 300 FOR J = F TO L: PRINT CHR
$ (A(J))" ";; NEXT : PRIN
T : REM CONTROLLING CHARA
CTERS
32 310 PRINT E$"*" CHR$ (1);: RE
M DL RAM "ON"
8D 320 FOR J = F TO L: PRINT CHR
$ (A(J))" ";; NEXT : PRIN
T : REM DL CHARACTERS
96 330 PRINT E$"*" CHR$ (0): REM
DL RAM "OFF"
37 340 IF L < > S THEN F = F + C
L:L = L + CL:R = R - CL:
GOTO 280
98 350 PRINT E$"@";: RETURN
48 1010 DATA 49,49,139,4,10,160,
10,96,10,32,28,2,0,0: RE
M a grave
88 1020 DATA 50,50,139,4,10,96,1
0,160,10,96,28,2,0,0: RE
M CIRCUMFLEX
34 1030 DATA 51,51,139,28,34,8,3
4,72,34,136,34,24,0,0: R
EM ACCUTE
C8 1040 DATA 52,52,139,28,34,136
,34,72,34,8,34,24,0,0: R
EM GRAVE
17 1050 DATA 53,53,139,28,34,72,
34,136,34,72,34,24,0,0:
REM CIRCUMFLEX
8F 1060 DATA 54,54,152,0,98,0,19
0,0,66,0,0,0,0,0: REM CI
RCUMFLEX
E3 1070 DATA 55,55,152,0,162,0,6
2,0,130,0,0,0,0,0: REM D
IERESIS
7E 1080 DATA 56,56,139,28,34,64,
34,128,34,64,34,28,0,0:
REM CIRCUMFLEX
87 1090 DATA 57,57,139,60,2,128,
2,64,2,0,60,2,0,0: REM G
RAVE
A8 1100 DATA 48,48,139,60,2,64,2
,128,2,64,60,2,0,0: REM
CIRCUMFLEX
38 1110 DATA 37,37,139,28,34,136
,34,8,34,136,34,24,0,0:
REM DIERESIS
28 1120 DATA 35,35,10,56,68,0,69
,0,71,0,68,0,0,0: REM CE
DILLA
24 1130 DATA 36,36,139,120,132,0
,133,0,135,0,132,72,0,0:
REM CEDILLA
2C 1140 DATA 38,38,152,0,0,64,16
0,0,160,64,0,0,0,0: REM
DEGREE

```

©

Amiga Disk-Based Fonts

Daniel L. Stockton

Expanding on a previous, related article (see "Amiga System Fonts" in the April 1987 issue of COMPUTE!), this article shows you how to load custom text fonts from disk and install them from Amiga BASIC.

In the Utilities drawer of the Amiga Workbench disk is a useful tool called Notepad. One attractive feature of this mini-word processor is its ability to use a variety of text fonts. This article explains how to use those same fonts—or any disk-based text font—in Amiga BASIC. Amiga BASIC does not provide any direct means of loading a custom font from disk. However, this can be accomplished by calling system routines which are used by the computer itself.

Where Are My Fonts?

The fonts used by Notepad are located in the *fonts* directory of the Workbench disk. You can use the FILES command to get a listing of that directory. Type this command in the BASIC output window and press Return:

files "workbench:fonts"

Each font in the fonts directory has its own subdirectory which contains the various sizes for that font. Font sizes are specified in units called *points*, which are equal to 1/72 inch. Thus, a 9-point font has characters 9/72 inch in size, and so on. The different text styles (italicized, boldface, and so on) are not stored in the font directory; these styles are generated by selec-

tively distorting the shapes found in the basic font file.

The program included with this article can select any of the disk-based fonts, with eight styles for each font. Table 1 lists the fonts, and Table 2 lists the various styles. Version 1.2 of the Amiga operating system adds a few new sizes to existing fonts.

Table 1: Fonts

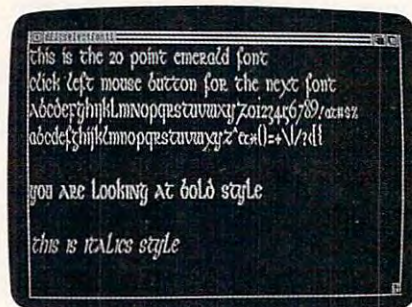
Fonts	Sizes
Topaz	8, 9, 11
Ruby	8, 12, 15
Diamond	12, 20
Opal	9, 12
Emerald	17, 20
Garnet	9, 16
Sapphire	14, 19

Table 2: Font Styles

0	plain text (Workbench default)
1	underline
2	boldface
3	boldface and underline
4	italics
5	italics and underline
6	italics and bold
7	italics, bold, and underline

Program 1 uses two system libraries named *graphics.library* and *diskfont.library*. In order for Amiga BASIC to use these libraries, it must have a file description of the library in a form which it understands. This form is called a *bmap file*. The *bmap file* is essentially a list of pointers that allow BASIC to access library routines.

Before you can run Program 1, you must make sure that the correct *bmap files* are present on the same



This article shows how to access disk-resident character fonts from Amiga BASIC. Two of the Amiga's seven fonts are shown in these photographs.

disk as Program 1. The first such file, *graphics.bmap*, is included in the BASICDemos drawer of the Amiga Extras disk. The second file, named *diskfont.bmap*, must be created.

If you have version 1.2 of the Amiga operating system (available as an inexpensive upgrade from any Amiga dealer), you can create *diskfont.bmap* quite easily. The BASICDemos disk for 1.2 contains a BASIC program named *ConvertFd*, as well as a directory named *FD1.2*. Run the *ConvertFd* program, using the file named *diskfont.lib_fd*. When the program is

finished, the disk contains diskfont.bmap. If you haven't obtained the 1.2 upgrade, type in and run Program 2, which reads values from DATA statements, checks them for accuracy, and creates diskfont.bmap on the current disk.

Before you run Program 1, make sure that both graphics.bmap and diskfont.bmap are on the same disk as Program 1. The location of these files is important. They must be either in the current directory or in the directory named LIBS on the disk used when you booted the system. The LIBS (LIBrarieS) directory is a good place for bmap files, since their purpose is to give you access to libraries. If you don't have the bmap files in the correct place, BASIC will stop with a *File not found* error when you run Program 1.

When you run the program, it displays the various fonts on the screen in different sizes and waits for you to click the mouse button before proceeding to the next font.

The most important part of Program 1 is contained in the subprogram named FontSelect, which appears at the end of the program. After you have tested Program 1 and have saved a copy, delete everything in the program except the FontSelect subprogram. Then save the subprogram under a new name as an ASCII file, so that it can be MERGED with other programs. To save a program in ASCII form, add the characters ,a to the end of a normal save command. For instance, to save the subprogram with the name FontSelect, you would type this command in the BASIC output window and press Return:

save "FontSelect",a

The FontSelect subprogram is invoked with a CALL statement in the main program. Three items of information are passed to FontSelect in the form of variables.

The first variable is a string named font\$. This string contains the name of the font you wish to use (Garnet, Ruby, and so on). If the string is a null string ("", a string containing no characters), only a style change occurs.

The second and third variables passed to FontSelect are numeric variables of the short integer type. The variable height% defines the

point size that you wish to use (see Table 1), and style% defines the style to use (Table 2).

Opening And Closing Libraries

A few additional statements are needed to prepare for the CALL to FontSelect, and to clean up afterward. In Program 1, the preparatory statements are grouped together at the beginning, immediately after the first two REMark statements. The DEFLNG statement causes all simple variables to default to the long integer type. (Note that this declaration is overridden by the short integer type specifier attached to height% and style%.)

The LIBRARY and DECLARE FUNCTION statements actually give you access to library routines. These statements should appear in the initialization section of the program, before the first CALL to the FontSelect subprogram.

When the program is about to terminate, you should take some additional steps to close the fonts and the libraries. The CALL to the CloseFont function closes any fonts that were previously opened. (A bug in version 1.1 of the operating system prevents this CALL from working correctly. Version 1.2 corrects the bug. If you have version 1.1, omit the line containing the CALL to CloseFont or put a REM in front of the line.)

The final CALL to FontSelect resets the font to the system font, Topaz. While not absolutely necessary, it's considered good manners for programs which change the computer environment to restore the original environment as closely as possible before terminating.

The LIBRARY CLOSE statement closes libraries that were previously opened. If you omit these final housekeeping chores, the computer may not crash, but the libraries will remain open, wastefully occupying memory which would otherwise be freed for other tasks.

The program module named TestSection uses another system routine named text when printing words in boldface and italicized styles. This method prevents the characters in those words from overlapping, as they would if you printed the words with PRINT.

Program 1: Amiga Disk-Based Fonts

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

```
REM Copyright 1987 COMPUTE! Publications, Inc.
REM All Rights Reserved
REM Program SelectFont
REM Provides for use of Amiga disk based fonts from Amiga Basic
PRINT "Copyright 1987 COMPUTE! Publications, Inc."
PRINT "All Rights Reserved"
DEFLNG a-z 'all variables default to long integer
'Include optional CHDIR command here (CHDIR ":BMAPS")
LIBRARY "diskfont.Library"
LIBRARY "graphics.Library"
DECLARE FUNCTION OpenDiskFont LIBRARY
DECLARE FUNCTION AskSoftStyle LIBRARY
'the above commands must be placed in the main body of your program
TestSection:
BREAK ON
ON BREAK GOSUB Housekeeping
READ t1$,t2$
FOR i=0 TO 6 'look at 7 fonts
READ font$,height%
CALL FontSelect(font$,height%,style%)
FOR style%=0 TO 4 STEP 2 'look at 3 styles each
CALL FontSelect("",0,style%)
IF style%=0 THEN
LOCATE 1,1:PRINT "This is the";
height%;"point ";font%;" font"
PRINT "Click left mouse button for the next font"
PRINT t1$:PRINT t2$
ELSEIF style%=2 THEN
a$="YOU ARE LOOKING AT BOLD STYLE"
PRINT
CALL text(WINDOW(8),SADD(a$),LEN(a$)):PRINT
ELSEIF style%=4 THEN
a$="THIS IS ITALICS STYLE"
PRINT
CALL text(WINDOW(8),SADD(a$),LEN(a$)):PRINT
END IF
NEXT style%
WaitForMouse:
IF NOT MOUSE(0) THEN WaitForMouse
CLS
'include the following CALL statement when using workbench 1.2
'as it closes fonts and frees memory
CALL CloseFont(WINDOW(8),fontptr)
NEXT i
Housekeeping:
CALL FontSelect("topaz",8,0) 'return to default system font
LIBRARY CLOSE
END
DATA ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789!@#%&
DATA abcdefghijklmnopqrstuvwxyz^&*()=+|\|/?<[{}&
```

```

DATA topaz,9,ruby,12,diamond,12,
opal,11<
DATA emerald,20,garnet,16,sapphi
re,19<
REM End Of TestSection<
<
SUB FontSeLect(font$,height%,sty
Le%) STATIC <
IF font$ <> "" THEN<
textAttr(0)=SADD(font$+".font"+C
HR$(0))<
textAttr(1)=height%*65536<
fontptr=OpenDiskFont(VARPTR(text
Attr(0)))<
IF fontptr THEN SetFont WINDOW(8
),fontptr<
END IF<
permitted%=AskSoftStyLe&(WINDOW(8
))<
CALL SetSoftStyle (WINDOW(8),sty
Le%,permitted%)<
END SUB<
<
<
<

```

Program 2: DiskFont.bmap Filemaker

```

REM Copyright 1987 COMPUTE! Publ
ications, Inc.<
REM All Rights Reserved<
' DiskFontMaker. Creates Diskfon
t.bmap file on disk.<
PRINT "Copyright 1987 COMPUTE! P
ublications, Inc.<
PRINT "All Rights Reserved"<
file$="Diskfont.bmap"<
READ fileSize,checksum<
PRINT "Checking data statements.
.."<
PRINT<
FOR j=1 TO fileSize<
READ a$<
a=VAL("&H"+a$)<
check=check+a<
NEXT j<
RESTORE DiskFontData<
IF check=checksum THEN GOTO OK<
PRINT "Typing error: Check DATA
statements."<
END<
<
OK:<
PRINT "Creating Diskfont.bmap fi
le..."<
ON ERROR GOTO CreationError<
OPEN file$ FOR OUTPUT AS #1<
FOR j=1 TO fileSize<
READ a$<
a=VAL("&H"+a$)<
PRINT #1,CHR$(a);<
NEXT j<
CLOSE<
PRINT "Finished"<
END<
<
CreationError:<
PRINT "Error ";ERR<
END<
<
DATA 34, 3196<
DiskFontData:<
DATA 4f,70,65,6e,44,69,73,6b<
DATA 46,6f,6e,74,00,ff,e2,09<
DATA 00,41,76,61,69,6c,46,6f<
DATA 6e,74,73,00,ff,dc,09,01<
DATA 02,00<
<

```

©

Atari NoDOS

Emmanuel Gendrano

Everyone knows it's impossible to save an Atari BASIC program to disk without DOS—or is it? Although you may not need to use it often, this program can be a real lifesaver on occasions where DOS becomes corrupted, or when you enter BASIC without first booting DOS.

One of the most fearsome messages an Atari BASIC programmer can receive is ERROR 130, the unknown device error. If this message appears when you attempt to save a BASIC program to disk, it usually means that the system doesn't recognize the disk drive, leaving you no way to save your work. This sometimes happens when misdirected POKES or other accidents corrupt the DOS (Disk Operating System) in memory. It can also occur if you forget to boot DOS when you turn on the computer.

Since the purpose of DOS is to perform disk operations, it seems logical that you can't save a BASIC program if DOS isn't present. But "Atari NoDOS" can perform this seemingly impossible task.

Using Atari NoDOS

Atari NoDOS consists of three short sections: one that formats a disk without DOS, one that saves a program without DOS, and a third that reloads a program saved in NoDOS format. Type in the entire

program and save it to disk. If you have a cassette drive, you may want to LIST the program to tape, as well. (If you have a cassette drive, you can always try to CSAVE a program when DOS is absent. However, Atari cassette drives are much less reliable than disk drives, and using NoDOS is also much faster than saving to cassette.)

Of course, if you're in a situation where you need Atari NoDOS, you won't be able to ENTER the program from disk. At this point, you can either ENTER it from cassette, if you have a cassette drive, or simply type the program from the magazine listing. The program is so short that it can be typed in only a few minutes.

If you wish to save the program to an already-formatted disk, you need only type lines 32760–32767 of Atari NoDOS. If you need to format a new disk before performing the save, you must type the formatting section of the program (lines 32755–32757) as well as the saving section (32760–32767). Of course, if you ENTER the program from cassette, all of the lines are present.

To save a program on an already-formatted disk, type GOTO 32760 and press RETURN. To format a disk and then save the program, type GOTO 32755 and press RETURN; the formatting routine jumps to the saving routine after the disk is formatted.

The saving routine prompts you to enter a single-density disk in drive 1 and press RETURN when you are ready. The disk should have enough room to store the program, but it need not be completely empty. Atari NoDOS doesn't save the program in the usual manner. Instead, it writes the program on disk beginning with sector 719 and proceeding to lower-numbered sectors. As you may know, single-density Atari disks contain 720 sectors, and program storage normally begins with lower-numbered sectors and works upward. By writing to the highest sectors on the disk, NoDOS minimizes the likelihood of overwriting information already on the disk. However, it does not check to see whether a given sector already contains information, so you should not attempt to use this program with a disk that is almost full.

The screen blanks out while Atari NoDOS saves the program. This is done to gain a little extra speed. Just as in a normal disk save, you will see the drive's busy light come on. When the screen is restored, the program displays a status number to indicate the success of the operation. If this number is anything other than 1, an error occurred; your disk drive manual explains the meaning of the error number.

If the save was successful, you should turn off the computer, insert a disk containing DOS, and turn the computer back on. Don't expect to see the program saved using NoDOS in the disk directory. It has been saved in a different format which doesn't appear in the directory.

To convert the program to normal form, you must now load the reloader section of Atari NoDOS. Again, if you saved the entire program to disk, simply reload it. Since the reloader is the first section of Atari NoDOS, you can start it by typing either RUN or GOTO 32740. When the program prompts you to do so, insert the disk on which you saved the program without DOS; then hit RETURN. Each line is displayed on the screen as it is entered into memory. If you wish to abort the process, simply press a key. When the process is complete, you

should immediately save the program in the usual way.

This program can be modified to work with double-density disks, as well. Change the first two POKES in line 32767 to POKE 776,0 and POKE 777,1. You may or may not have to change the same POKES in the reloading routine, depending on your drive.

If you get consistent errors when using the saving routine, your drive may be misaligned—a problem which requires a professional cure. However, you may be able to save the program by changing Atari NoDOS to write to the lowest sectors on the disk rather than the highest. The lower sectors are much easier to read, even for a misaligned drive. Note that this method can be used *only with a completely blank disk*, since it is certain to destroy at least the first file on the disk. To make this modification, change the statement SEC=719 to SEC=1 in lines 32761 and 32742. You must also change the statement SEC=SEC-1 to SEC=SEC+1 in lines 32767 and 32752. These changes cause the program to begin at sector 1 and use higher-numbered sectors as the save proceeds.

Atari NoDOS

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

```

KP 0 REM COPYRIGHT 1987 COMP
    UTE! PUBLICATIONS, INC.
    ALL RIGHTS RESERVED
ND 1 PRINT CHR$(125);"COPYRI
    GHT 1987":PRINT "COMPUT
    E! PUBLICATIONS, INC."
DL 2 PRINT "ALL RIGHTS RESER
    VED."
BF 3 FOR X=1 TO 1000:NEXT X
IP 32739 REM -----
HG 32740 REM NoDOS Saves a
    BASIC program from
    memory to disk wit
    hout any DOS presen
    t. 10:24AM/7-18-86
ND 32741 GRAPHICS 0:CLR:DIM
    A$(1):? :? "NoDOS
Reloader":? "Inse
    rt disk in drive 1
    ":TRAP 32740
AD 32742 INPUT A$:BUF=1536:S
    EC=719:I=128:POKE 8
    2,0:POKE 83,39
BC 32743 POKE 842,12:L=0:? "
    {CLEAR}{TAB}HIT ANY
    KEY TO ABORT{DOWN}
    ":POKE 766,1
FN 32744 GOSUB 32749:? CHR$(
    D):L=L+1:IF D=155
    THEN 32747
DC 32745 IF PEEK(764)<>255 T
    HEN POKE 82,2:POKE
    766,0:POKE 764,255:
    END
  
```

```

ED 32746 GOTO 32744
DP 32747 POKE 766,0:IF L=1 T
    HEN POKE 82,2:END
HC 32748 ? "{DOWN}G.32743":P
    OKE 842,13:POSITION
    0,0:STOP
BK 32749 IF I=128 THEN GOSUB
    32751
NI 32750 D=PEEK(BUF+I):I=I+1
    :RETURN
JB 32751 POKE 768,49:POKE 76
    9,1:POKE 770,82:POK
    E 771,64:POKE 772,A
    SC(CHR$(BUF)):POKE
    773,INT(BUF/256):PO
    KE 776,128
NK 32752 POKE 777,0:POKE 778
    ,ASC(CHR$(SEC)):POK
    E 779,INT(SEC/256):
    D=USR(ADR("hLYE")):
    SEC=SEC-1:I=0:RETUR
    N
IN 32754 REM -----
NH 32755 ? :? "NoDOS/Format
E drive #1.":? "Ent
    er 'Y' when ready "
    ;:CLR:DIM A$(1):IN
    PUT A$
ED 32756 IF A$<>"Y" THEN 327
    60
BE 32757 POKE 768,49:POKE 76
    9,1:POKE 770,33:POK
    E 771,0:POKE 774,25
    5:D=USR(ADR("hLYE"))
    )
JB 32759 REM -----
NB 32760 ? :? "NoDOS/Save
E
    Hit RETURN to star
    t":CLR:DIM B$(25
    5),A$(127):INPUT A$
    :TRAP 40000:CLOSE #
    1:POKE 82,0
CN 32761 POKE 83,39:BUF=ADR(
    B$):I=1:SEC=719:OPE
    N #1,13,0,"E":POKE
    752,1:POKE 559,0:A
    =PEEK(136)+256*PEEK
    (137)
HG 32762 ? "{CLEAR}":L=PEEK
    (A)+256*PEEK(A+1):L
    IST L:POSITION 0,1:
    INPUT #1,A$:A$(LEN(
    A$)+1)=CHR$(155)
EN 32763 B$(I)=A$:I=LEN(B$)+
    1:A=A+PEEK(A+2):IF
    L<32739 AND I<=128
    THEN 32762
GP 32764 IF I>128 THEN GOSUB
    32766:B$(I)="#":B$
    =B$(129):I=LEN(B$):
    B$(I)="":IF PEEK(77
    1)=1 AND L<32739 TH
    EN 32762
AP 32765 B$(129)="#":B$(I)=C
    HR$(155):B$(I+1)=B$
    (I):GOSUB 32766:POK
    E 82,2:GRAPHICS 0:?
    "{BELL}STATUS - ";
    PEEK(771):END
PJ 32766 POKE 768,49:POKE 76
    9,1:POKE 770,87:POK
    E 771,128:POKE 772,
    ASC(CHR$(BUF)):POKE
    773,INT(BUF/256)
ME 32767 POKE 776,128:POKE 7
    77,0:POKE 778,ASC(C
    HR$(SEC)):POKE 779,
    INT(SEC/256):D=USR(
    ADR("hLYE")):SEC=SE
    C-1:RETURN
  
```

©

Fast Fractal Landscapes For IBM

Paul W. Carlson

It doesn't take an expensive mainframe computer to create realistic-looking fractal landscapes. Using the program included in this article, you can create your own landscapes on any IBM PC or compatible computer with a color graphics adapter.

Some recent magazine articles about fractals have included pictures of computer-generated landscapes so realistic that they appear to be photographs of nature itself. Although "Fast Fractal Landscapes" doesn't produce displays that resemble photographs, it does create realistic landscapes in the sense that all of the scenes resemble nature. And best of all—it creates them in about half a minute.

To get started, type in and save the program listed on the following page. Before you run the program, make sure you have a disk in the active drive with at least 20,000 bytes free. Now run the program. The program will create a file on the disk called `FRACLAND.COM`. To run `FRACLAND.COM`, first type `SYSTEM` to get out of `BASIC`. At the `DOS` prompt, type `FRACLAND` and press Enter. You'll get a message that elevations are being computed. The plot will begin a few seconds later. When a plot is complete, press any key except the `Q` key to start another plot. Pressing the `Q` key exits the program.

The program creates the displays so fast that you might think that all the landscapes are stored within the program. The truth is,

however, that the program is creating them randomly. The program uses what might be described as "controlled randomness" to create over 30,000 different landscapes. If you don't get a landscape that you like when you first run the program, keep trying—you'll soon get one that you like.

Some of the plots may be almost all land and others may be almost all water. In fact, it's possible that some might be all water. Overall, the amount of land and water should be close to equal. The plots have the vertical scale exaggerated to keep some separation between the lines that form the land. This, along with the colors used, produces landscapes that look like rugged terrain in winter at sunset.

How It Works

The program employs a technique that is often used when dealing with fractals: repeatedly subjecting an object (which can be a line, shape, or almost anything) to a series of operations at an ever-decreasing scale. In the language of fractals, the initial object is called the *initiator* and the series of operations is called the *generator*.

The initiator in this program is a square representing a flat area of land initially at sea level, or zero elevation. Here is a description of the operations performed on this square and on all subsequent squares:

1. A new elevation is computed for the midpoint of each side of the square by averaging the eleva-

tions of the corners on either side of the midpoint.

2. Each new midpoint elevation is increased or decreased by a random amount. The random amount is computed so that its maximum absolute value is proportional to the length of the side containing the midpoint. This keeps the height of the landforms in natural proportion to their breadth.

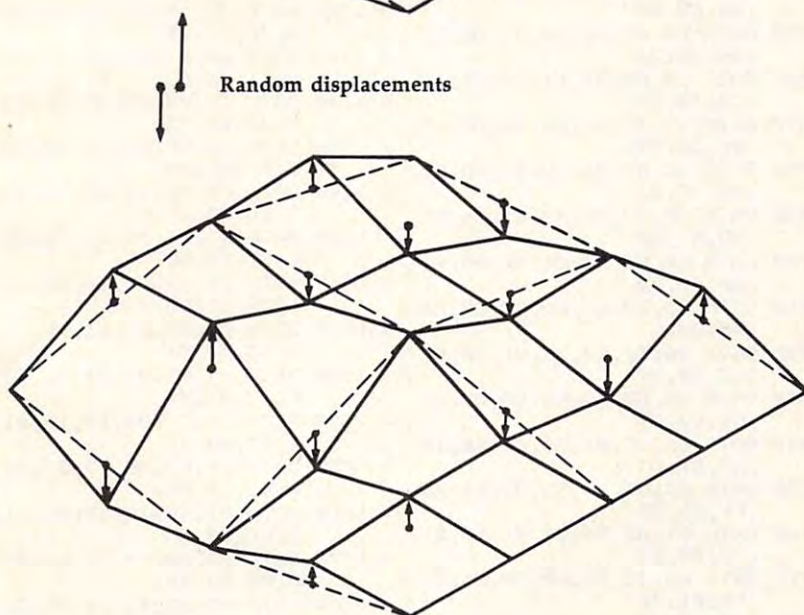
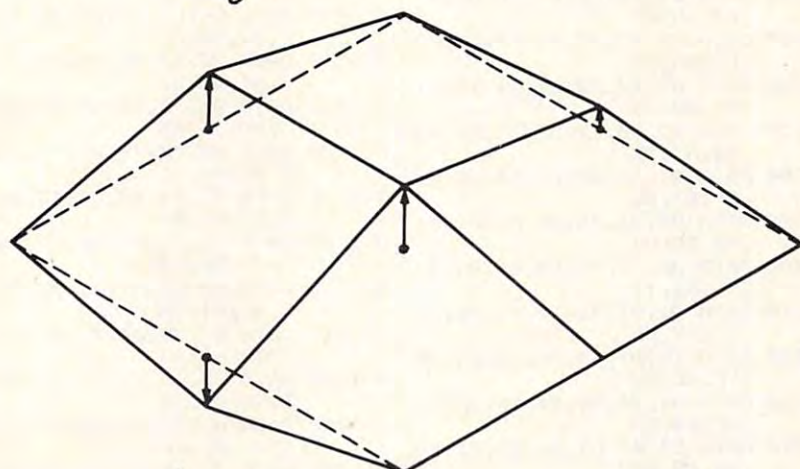
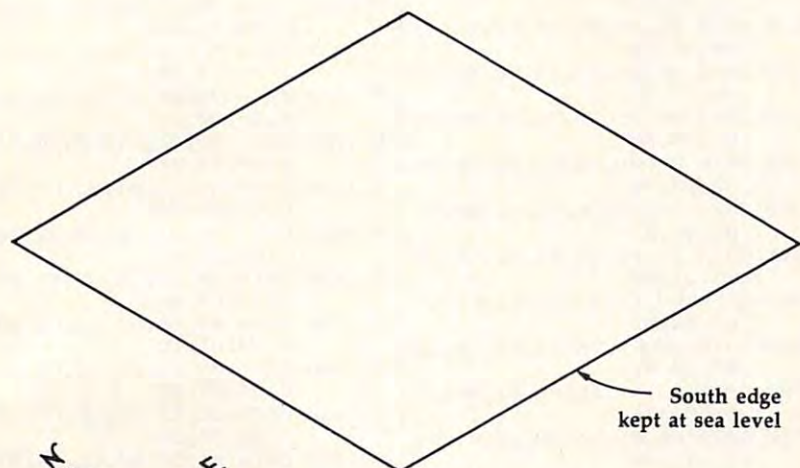
3. The elevation of the point at the center of the square is computed in the same manner using the elevations of a pair of diagonally opposite corners, the pair used being chosen randomly.

4. The square is subdivided into four squares, the midpoints and center of the old square becoming corners of the new squares.

5. The process is then repeated for each of the new squares. Only when all the squares at any level of subdivision have been processed does the computation move on to the next level. When the desired level of subdivision has been accomplished, the process stops. The process of subdivision is illustrated in the accompanying figure.

If the elevations were to be plotted without further processing, the resultant landscape would resemble a scene on the moon with very jagged mountain peaks. To make the landscape more earthlike, the elevations are smoothed by averaging each elevation with the adjacent east and west elevations (north being the top of the screen). This gives the landscapes a more

Landscape Subdivision



rounded, eroded appearance.

The display is created by plotting cross sections from west to east beginning with the southernmost cross section. Each cross section is plotted by connecting the north sides of adjacent squares end to end and removing hidden lines. Before a line segment on a cross section is plotted, the program checks if the segment is going into or coming out of the water. If it is, the segment is adjusted to end at the water level (elevation zero). Any negative elevations are set to elevation zero. Segments with both endpoints at elevation zero are plotted as water. The elevations at the south edge of the plot are kept at sea level to keep the first cross section from slicing through any mountains.

Fast Fractal Landscapes

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

```

EG 1 ' Program to create FRACLAN
    D.COM
OH 2 '
NC 3 ' Copyright 1987 Compute! P
    ublications, Inc.
DL 4 ' All Rights Reserved
HP 5 CLS:PRINT "Copyright 1987 C
    ompute! Publications, Inc."
PN 6 PRINT "All Rights Reserved"
DO 10 OPEN "FRACLAND.COM" FOR OU
    TPUT AS 1
NA 20 PRINT:PRINT "CREATING FRACL
    AND.COM FILE, PLEASE WAIT.
    .-";
JM 30 PRINT#1,CHR$(&HE9);CHR$(&H
    CA);CHR$(&H44);
BN 40 FOR N=1 TO 17577:PRINT#1,C
    HR$(N);:NEXT
AD 50 T=0:FOR J=1 TO 2049:READ A
    $:N=VAL("&H"+A$)
FP 60 T=T+N:PRINT#1,CHR$(N);:NEX
    T:CLOSE 1:PRINT
HN 70 IF T=202520! THEN PRINT"FR
    ACLAND.COM SUCCESSFULLY CR
    EATED!":END
LP 80 PRINT CHR$(7);"***** ERROR
    IN DATA STATEMENTS *****"
    :END
OA 90 DATA 00,00,00,00,00,00,00,
    00,00,43
PF 100 DATA 6F,6D,70,75,74,69,6E
    ,67,20,65
MD 110 DATA 6C,65,76,61,74,69,6F
    ,6E,73,2E
HJ 120 DATA 2E,2E,24,EB,87,07,A1
    ,2B,01,A3
KF 130 DATA 2D,01,C7,06,2F,01,00
    ,00,EB,54
HI 140 DATA 07,BB,02,00,50,EB,41
    ,07,83,C4
NG 150 DATA 02,8D,16,85,45,B4,09
    ,CD,21,C7
FO 160 DATA 06,27,01,00,00,8B,1E
    ,27,01,D1
OP 170 DATA E3,C7,87,31,01,C8,00
    ,FF,06,27
GD 180 DATA 01,81,3E,27,01,40,01
    ,7C,EB,C7
    
```

HG 190	DATA 06,0F,01,40,00,C7,06,09,01,00	,82,00,F7	OH 1000	DATA 01,02,7D,1A,C7,06,25,01,00,00	
NC 200	DATA 00,A1,0F,01,A3,0D,01,E9,0B,02	PG 600	DATA 2E,13,01,8B,F8,B9,89,B1,03,EB	EN 1010	DATA C7,06,29,01,01,00,83,3E,15,01
LK 210	DATA C7,06,07,01,00,00,A1,09,01,03	KB 610	DATA 5A,FF,36,0F,01,E8,1F,05,83,C4	PM 1020	DATA 40,75,0D,83,3E,05,01,00,74,06
DD 220	DATA 06,0D,01,D1,F8,A3,13,01,A1,0F	BM 620	DATA 02,8B,C8,D1,E0,03,C1,8B,1E,0B	DJ 1030	DATA FF,06,15,01,EB,C3,C7,06,17,01
EK 230	DATA 01,A3,0B,01,E9,DB,01,A1,07,01	BJ 630	DATA 01,D1,E3,8B,C8,B8,82,00,F7,2E	QP 1040	DATA 00,00,A1,17,01,8B,C8,D1,E0,D1
PH 240	DATA 03,06,0B,01,D1,F8,A3,11,01,B8	DF 640	DATA 09,01,8B,F0,8B,80,B1,03,8B,1E	KE 1050	DATA E0,03,C1,A3,21,01,0B,C9,75,03
OP 250	DATA 82,00,F7,2E,09,01,8B,F0,FF,36	MD 650	DATA 07,01,D1,E3,8B,D0,B8,82,00,8B	BD 1060	DATA E9,10,01,83,F9,3F,7C,03,E9,0B
MN 260	DATA 0F,01,E8,80,06,83,C4,02,8B,1E	NF 660	DATA F2,F7,2E,0D,01,8B,F8,8B,81,B1	OP 1070	DATA 01,83,3E,15,01,40,7C,03,E9,FE
HJ 270	DATA 07,01,D1,E3,8B,C8,8B,80,B1,03	ML 670	DATA 03,03,C6,03,C1,99,2B,C2,D1,F8	AG 1080	DATA 00,83,3E,05,01,00,75,03,E9,8A
CB 280	DATA 8B,1E,0B,01,D1,E3,03,80,B1,03	PD 680	DATA 8B,1E,11,01,D1,E3,8B,C8,B8,82	JL 1090	DATA 00,8B,82,00,F7,E9,8B,F0,A1,15
HG 290	DATA 99,2B,C2,D1,F8,03,C1,8B,1E,11	JF 690	DATA 00,F7,2E,13,01,8B,F0,89,8B,B1	KE 1100	DATA 01,D1,E0,03,F0,8B,B8,C3,03,03
GE 300	DATA 01,D1,E3,89,80,B1,03,8B,82,00	FA 700	DATA 03,A1,0F,01,01,06,07,01,01,06	GI 1110	DATA BC,2F,03,0B,FF,7E,27,8B,84,B3
ID 310	DATA F7,2E,0D,01,8B,F0,FF,36,0F,01	AE 710	DATA 0B,01,83,3E,0B,01,40,7F,03,E9	CN 1120	DATA 03,03,84,B1,03,89,46,FE,0B,C0
PH 320	DATA E8,46,06,83,C4,02,8B,1E,07,01	FO 720	DATA 1B,FE,A1,0F,01,01,06,09,01,01	KA 1130	DATA 7D,18,8B,C8,D1,E0,D1,E0,03,C1
DE 330	DATA D1,E3,8B,C8,8B,80,B1,03,8B,1E	IC 730	DATA 06,0D,01,83,3E,0D,01,40,7F,03	OH 1140	DATA 99,8B,8C,B3,03,03,8C,B1,03,2B
PD 340	DATA 0B,01,D1,E3,03,80,B1,03,99,2B	AC 740	DATA E9,EB,FD,D1,3E,0F,01,83,3E,0F	FL 1150	DATA CF,F7,D9,E9,AD,00,8B,82,00,F7
BD 350	DATA C2,D1,F8,03,C1,8B,1E,11,01,D1	GN 750	DATA 01,02,7C,03,E9,CE,FD,C7,06,15	QP 1160	DATA 2E,17,01,8B,F0,A1,15,01,D1,E0
PD 360	DATA E3,89,80,B1,03,83,3E,07,01,00	IB 760	DATA 01,00,00,C7,06,17,01,01,00,8B	CM 1170	DATA 03,F0,8B,84,35,04,03,84,33,04
PE 370	DATA 74,44,8B,36,07,01,D1,E6,FF,36	NH 770	DATA 82,00,F7,2E,17,01,8B,F0,A1,15	GJ 1180	DATA 0B,C0,7F,03,E9,94,00,8B,BC,B3
HG 380	DATA 0F,01,E8,0B,06,83,C4,02,8B,C8	IL 780	DATA 01,D1,E0,03,F0,8B,84,2F,03,03	PF 1190	DATA 03,03,BC,B1,03,78,03,E9,87,00
NH 390	DATA B8,82,00,F7,2E,0D,01,8B,D8,8B	QL 790	DATA 84,33,04,03,84,B1,03,99,89,03	IJ 1200	DATA 8B,C7,8B,C8,D1,E0,D1,E0,03,C1
BD 400	DATA 82,00,F7,2E,09,01,8B,8B,B1,03	IF 800	DATA 00,F7,F9,89,84,B1,03,FF,06,17	HB 1210	DATA 99,8B,8C,B3,03,2B,8C,35,04,2B
MA 410	DATA 8B,D8,8B,80,B1,03,03,C7,99,2B	DD 810	DATA 01,83,3E,17,01,40,7C,CF,FF,06	KK 1220	DATA 8C,33,04,03,8C,B1,03,EB,64,8B
PF 420	DATA C2,D1,F8,03,C1,8B,C8,8B,82,00	AB 820	DATA 15,01,83,3E,15,01,40,7E,BE,EB	NF 1230	DATA 82,00,F7,2E,17,01,8B,F0,A1,15
MA 430	DATA F7,2E,13,01,8B,D8,89,8B,B1,03	HA 830	DATA A1,04,8B,04,00,50,EB,8E,04,83	EM 1240	DATA 01,D1,E0,03,F0,83,BC,2F,03,00
JE 440	DATA 8B,36,0B,01,D1,E6,FF,36,0F,01	OB 840	DATA C4,02,C7,06,05,01,00,00,B9,03	FH 1250	DATA 7E,1E,83,BC,B1,03,00,7D,17,8B
PH 450	DATA E8,C4,05,83,C4,02,8B,C8,8B,82	NP 850	DATA 00,89,0E,03,01,8B,C7,00,50,2B	GM 1260	DATA 84,B1,03,8B,C8,D1,E0,D1,E0,03
AC 460	DATA 00,F7,2E,0D,01,8B,D8,8B,82,00	NM 860	DATA C0,50,8B,01,00,50,2B,C0,50,EB	NC 1270	DATA C1,99,8B,8C,2F,03,2B,8C,B1,03
CF 470	DATA F7,2E,09,01,8B,8B,B1,03,8B,D8	KI 870	DATA 15,03,83,C4,0B,8B,C7,00,50,8B	AE 1280	DATA EB,2F,8B,82,00,F7,2E,17,01,8B
EI 480	DATA 8B,80,B1,03,03,C7,99,2B,C2,D1	BP 880	DATA 3C,01,50,8B,01,00,50,8B,3C,01	EI 1290	DATA F0,A1,15,01,D1,E0,03,F0,83,BC
BC 490	DATA F8,03,C1,8B,C8,8B,82,00,F7,2E	FK 890	DATA 50,EB,FF,02,83,C4,0B,8B,01,00	IC 1300	DATA 33,04,00,7E,1E,83,BC,B1,03,00
F1 500	DATA 13,01,8B,D8,89,8B,B1,03,EB,B1	OH 900	DATA 50,8B,3C,01,50,8B,01,00,50,2B	SD 1310	DATA 7D,17,8B,84,B1,03,8B,C8,D1,E0
ID 510	DATA 05,AB,01,74,60,FF,36,0F,01,EB	OE 910	DATA C0,50,EB,EA,02,83,C4,0B,8B,C7	OB 1320	DATA D1,E0,03,C1,99,2B,8C,33,04,F7
JP 520	DATA 7F,05,83,C4,02,8B,C8,D1,E0,03	BD 920	DATA 00,50,8B,3C,01,50,8B,C7,00,50	HJ 1330	DATA F9,01,06,21,01,83,3E,05,01,00
JL 530	DATA C1,8B,1E,0B,01,D1,E3,8B,C8,8B	LB 930	DATA 2B,C0,50,EB,D5,02,83,C4,0B,8B	QJ 1340	DATA 74,31,8B,1E,15,01,D1,E3,8B,82
HA 540	DATA 82,00,F7,2E,0D,01,8B,F8,8B,81	FC 940	DATA 06,27,01,00,00,8B,1E,27,01,D1	PD 1350	DATA 00,F7,2E,17,01,8B,F0,8B,80,B1
NM 550	DATA B1,03,8B,1E,07,01,D1,E3,8B,D0	AD 950	DATA E3,C7,87,31,01,C8,00,FF,06,27	CM 1360	DATA 03,8B,1E,15,01,D1,E3,8B,C8,8B
FA 560	DATA B8,82,00,8B,FA,F7,2E,09,01,89	GH 960	DATA 01,81,3E,27,01,40,01,7C,EB,C7	ED 1370	DATA 82,00,F7,2E,17,01,8B,F0,8B,80
BM 570	DATA 7E,FE,8B,F8,8B,81,B1,03,03,46	BM 970	DATA 06,15,01,00,00,83,3E,15,01,40	DE 1380	DATA B3,03,03,C1,99,2B,C2,D1,F8,EB
FJ 580	DATA FE,03,C1,99,2B,C2,D1,F8,8B,1E	KK 980	DATA 7E,03,E9,5E,02,C7,06,05,01,00	PJ 1390	DATA 13,8B,1E,15,01,D1,E3,8B,82,00
LN 590	DATA 11,01,D1,E3,8B,C8,8B,82,00,F7	KC 990	DATA 00,C7,06,27,01,00,00,83,3E,27	HF 1400	DATA F7,2E,17,01,8B,F0,8B,82,00

	B, 80, B1, 03	DF 1810	DATA 6C, 8B, 7E, 04, D1, E7, B1, C7, 31, 01
FP 1410	DATA A3, 23, 01, 0B, C0, 7D, 06, C7, 06, 23	GM 1820	DATA 8B, 05, 39, 46, 06, 7D, 1F, 8B, 46, 06
OM 1420	DATA 01, 00, 00, 83, 3E, 23, 01, 00, 75, 0C	DD 1830	DATA 89, 05, 83, 3E, 03, 01, 03, 75, 07, 83
DH 1430	DATA 83, 3E, 25, 01, 00, 75, 05, 8B, 01, 00	BL 1840	DATA 3E, 05, 01, 00, 75, 0C, F7, 76, 04, FF
DE 1440	DATA EB, 03, 8B, 03, 00, A3, 03, 01, A1, 23	FK 1850	DATA 76, 06, E8, F1, 00, 83, C4, 04, 83, 7E
PN 1450	DATA 01, A3, 25, 01, A1, 21, 01, A3, 1D, 01	FH 1860	DATA FA, 00, 74, 08, 8B, 46, F2, 01, 46, 06
JM 1460	DATA A1, 15, 01, D1, E0, 03, 06, 05, 01, 03	GP 1870	DATA EB, 06, 8B, 46, F6, 01, 46, 04, 83, 7E
FJ 1470	DATA 06, 23, 01, 40, A3, 1F, 01, 83, 3E, 29	DI 1880	DATA F0, 00, 7D, 08, 8B, 46, E0, 01, 46, F0
PD 1480	DATA 01, 00, 74, 15, 8B, 01, 00, A3, 1D, 01	IE 1890	DATA EB, 1A, 8B, 46, F8, 01, 46, F0, 83, 7E
JJ 1490	DATA A3, 19, 01, A1, 1F, 01, A3, 1B, 01, C7	HB 1900	DATA FA, 00, 74, 08, 8B, 46, F6, 01, 46, 04
FD 1500	DATA 06, 29, 01, 00, 00, 83, 3E, 15, 01, 00	LD 1910	DATA EB, 06, 8B, 46, F2, 01, 46, 06, 4E, 0B
LH 1510	DATA 75, 07, 83, 3E, 05, 01, 00, 74, 16, FF	BE 1920	DATA F6, 75, 90, 5E, 5F, 8B, E5, 5D, C3, 55
MC 1520	DATA 36, 1F, 01, FF, 36, 1D, 01, FF, 36, 1B	CM 1930	DATA 8B, EC, 83, EC, 02, E8, 1E, 00, 8B, 4E
HN 1530	DATA 01, FF, 36, 19, 01, E8, 7B, 00, 83, C4	DE 1940	DATA 04, 49, 23, C8, 89, 4E, FE, E8, 12, 00
BD 1540	DATA 08, A1, 1D, 01, A3, 19, 01, A1, 1F, 01	FM 1950	DATA 3C, 7F, 73, 07, 8B, 46, FE, F7, D8, EB
KN 1550	DATA A3, 1B, 01, FF, 06, 17, 01, 83, 3E, 17	BL 1960	DATA 03, 8B, 46, FE, 8B, E5, 5D, C3, B9, 08
JE 1560	DATA 01, 40, 7D, 03, E9, F3, FD, 83, 3E, 05	OG 1970	DATA 00, A1, 2D, 01, 33, D2, A9, 02, 00, 74
JH 1570	DATA 01, 00, 75, 05, 8B, 01, 00, EB, 02, 2B	FO 1980	DATA 02, B2, 01, A9, 04, 00, 74, 02, B6, 01
KH 1580	DATA C0, A3, 05, 01, FF, 06, 27, 01, E9, AE	NK 1990	DATA 32, D6, D0, EA, D1, DB, E2, E8, A3, 2D
PL 1590	DATA FD, C7, 06, 03, 01, 02, 00, C7, 06, 1B	KP 2000	DATA 01, A3, 2F, 01, C3, 55, 8B, EC, 8B, 46
DF 1600	DATA 01, 81, 00, FF, 36, 1B, 01, 8B, 3B, 01	EO 2010	DATA 04, CD, 10, 8B, E5, 5D, C3, B4, 0F, CD
FN 1610	DATA 50, FF, 36, 1B, 01, 8B, 01, 00, 50, EB	JD 2020	DATA 10, B7, 07, 3C, 03, 7E, 06, 3C, 07, 74
HO 1620	DATA 27, 00, 83, C4, 08, FF, 06, 1B, 01, 81	KJ 2030	DATA 02, 32, FF, 8B, 00, 06, 33, C9, BA, 4F
AD 1630	DATA 3E, 1B, 01, C7, 00, 7C, DE, EB, E0, 01	LI 2040	DATA 18, CD, 10, 8B, 00, 02, 33, D2, 33, DB
EB 1640	DATA 0B, C0, 75, 03, E9, 1B, FA, 8B, 02, 00	EJ 2050	DATA CD, 10, C3, FB, B4, 00, CD, 1A, 89, 16
EG 1650	DATA 50, EB, 5F, 01, 83, C4, 02, EB, 65, 01	HI 2060	DATA 2B, 01, B9, 08, 00, A1, 2B, 01, 33, D2
IG 1660	DATA C3, 55, 8B, EC, 83, EC, 12, 57, 56, 8B	OB 2070	DATA A9, 02, 00, 74, 02, B2, 01, A9, 04, 00
LJ 1670	DATA C7, 00, 2B, 46, 06, 89, 46, 06, 8B, C7	CA 2080	DATA 74, 02, B6, 01, 32, D6, D0, EA, D1, DB
MH 1680	DATA 00, 2B, 46, 0A, 89, 46, 0A, 8B, 76, 08	FB 2090	DATA E2, EB, A3, 2B, 01, C3, 55, 8B, EC, 57
GO 1690	DATA 2B, 76, 04, 39, 46, 06, 7D, 28, 89, 46	EN 2100	DATA 56, A1, 03, 01, B4, 0C, 8B, 4E, 06, 8B
IP 1700	DATA F4, 8B, 46, 06, 89, 46, 0A, 8B, 46, F4	ED 2110	DATA 56, 04, CD, 10, 5E, 5F, 8B, E5, 5D, C3
OC 1710	DATA 89, 46, 06, 8B, 46, 08, B9, 46, F4, 8B	OP 2120	DATA B4, 07, CD, 21, 3C, 51, 74, 07, 3C, 71
PB 1720	DATA 46, 04, 89, 46, 08, 8B, 46, F4, 89, 46	BA 2130	DATA 74, 03, 33, C0, C3, 8B, 01, 00, C3
OG 1730	DATA 04, C7, 46, F6, FF, FF, EB, 05, C7, 46		
BP 1740	DATA F6, 01, 00, 8B, 46, 06, 2B, 46, 0A, 89		
DG 1750	DATA 46, FC, C7, 46, F2, FF, FF, C7, 46, FA		
IF 1760	DATA 00, 00, 3B, C6, 7E, 10, C7, 46, FA, 01		
JM 1770	DATA 00, 89, 76, F4, 8B, F0, 8B, 46, F4, 89		
CO 1780	DATA 46, FC, 8B, 46, FC, D1, E0, 89, 46, EE		
KA 1790	DATA 8B, 46, FC, 2B, C6, D1, E0, 89, 46, F8		
QI 1800	DATA 8B, 46, EE, 2B, C6, 89, 46, F0, 46, EB		

Atari XL/XE Super Editor

Rhett Anderson

This clever utility adds a variety of new features that improve the screen editor on Atari XL and XE computers. BASIC memory is unaffected.

Atari eight-bit computers are equipped with an exceptional screen editor. As you edit your BASIC programs, you can clear the screen, insert characters and lines, delete characters and lines, and move the cursor up, down, left, and right.

When Atari engineers developed the 1200XL computer, they used its new function keys to add even more editing features: one-key cursor control, cursor home, cursor to the bottom right corner, cursor to the left margin, cursor to the right margin, and a quick way to turn off the keyclick. Unfortunately, when the function keys were omitted in the newer Atari computers—the 600XL, 800XL, and the XE models—so were the new features. Fortunately, the code for these keys remains in the operating system. "Atari XL/XE Super Editor" resurrects this code, adding new editing power to the Atari 600XL, 800XL, 65XE, and 130XE.

Super Editor is written in machine language but is listed in the form of a BASIC loader. Since the DATA statements include machine language and key definitions which must be typed accurately, be sure to use the "Automatic Proofreader" found elsewhere in this issue to enter the program. Be sure to save a copy of the program before running it, since the BASIC part of the program is erased from memory after it is run.

Normal And Super Modes

After you have saved a copy of the program to tape or disk, type RUN. In a few seconds, you will see a message indicating that Super Editor is enabled. At this point, you have no new editing features. Super Editor has two modes of operation: normal and super. Whenever you run Super Editor, you'll be in the normal mode of operation, where all keystrokes act as they normally would. Change the mode with the HELP key, which acts as a toggle. Press it once to enter super mode, and again to return to normal mode. To help you tell the difference between the two modes, the screen color is changed to a darker blue in super mode. Here is a list of the new key assignments available in super mode:

CTRL-5 toggle keyclick
 CTRL-6 cursor home
 CTRL-7 cursor bottom left
 CTRL-8 cursor to left margin
 CTRL-9 cursor to right margin

In addition, you no longer need to press CTRL to cursor up, down, left, or right. To obtain the standard assignments of these keys (-, =, +, and *), press CTRL and the appropriate key.

None of these new assignments are available in normal mode, but normal mode does retain the keyclick value that you last used in super mode. If you turn the keyclick off in super mode, it remains off when you return to normal mode.

If you wish to disable Super Editor completely, press SYSTEM RESET. Type ?USR(1664) or DUMMY=USR(1664) in direct or program mode to reenable it.

Super Editor requires the top half of page 6 for its program and the bottom three quarters of page 1 (the stack) for the Super Editor key-

board table. These memory areas are not used by BASIC, but some utilities use them. A conflict could result in a locked-up computer, so be sure to test Super Editor with any other utilities that you may use before beginning any important work.

Atari XL/XE Super Editor

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

```

LA 1 REM COPYRIGHT 1987 COMP
    UTE! PUBLICATIONS, INC.
    {4 SPACES}ALL RIGHTS RE
    SERVED
EB 2 PRINT CHR$(125):PRINT "
    COPYRIGHT 1987"
AD 3 PRINT "COMPUTE! PUBLICA
    TIONS, INC."
AP 4 PRINT "ALL RIGHTS RESER
    VED"
BJ 5 FOR X=1 TO 1200:NEXT X
KF 10 FOR MEMORY=1664 TO 177
    4
BH 20 READ ML:POKE MEMORY,ML
    :CHECK=CHECK+ML:NEXT M
    EMORY
CF 30 IF 12623<>CHECK THEN P
    RINT "ERROR IN TYPING
    DATA STATEMENTS.":END
MC 40 DUMMY=USR(1664):? "
    {CLEAR}Atari XL/XE STU
    TE Editor Enabled.
    {5 SPACES}HELP key tog
    gles Super Editor.":NE
    W
PF 1010 DATA 104,162,0,189,8
    1
HO 1020 DATA 251,157,0,1,232
    ,224
CJ 1030 DATA 192,208,245,162
    ,12,189
JJ 1040 DATA 209,6,168,189,2
    22,6
EK 1050 DATA 153,0,1,202,16,
    243
JH 1060 DATA 162,6,160,166,1
    69,7
JD 1070 DATA 76,92,228,173,2
    20,2
IJ 1080 DATA 201,17,208,32,1
    69,0
IH 1090 DATA 141,220,2,173,2
    08,6
FB 1100 DATA 73,1,141,208,6,
    170
PI 1110 DATA 189,235,6,133,1
    21,189
IM 1120 DATA 237,6,133,122,1
    69,4
GH 1130 DATA 77,198,2,141,19
    8,2
NE 1140 DATA 76,98,228,1,6,7
OI 1150 DATA 14,15,134,135,1
    42,143
DC 1160 DATA 157,155,179,181
    ,176,30
CO 1170 DATA 31,28,29,43,42,
    45
CE 1180 DATA 61,137,142,143,
    144,145
ML 1190 DATA 0,81,1,251
  
```

COMPUTE! Disk Subscriptions

COMPUTE! Disks are available for the following computers:

- Apple II series
- Commodore 64 and 128
- Atari 400/800 /XL/XE
- IBM PC and PCjr

Each error-free disk contains all the programs from the previous three issues of COMPUTE!. With a disk subscription, you'll receive one disk—for the machine you specify—every three months. To subscribe, call toll free **800-247-5470** (in Iowa 800-532-1272).

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."

Resave

Amy Galtman

This convenient utility for the Commodore 64 lets you resave a program without fear of the infamous save-with-replace bug.

You have loaded a BASIC program from a disk and have decided that it needs a change or two. After editing, it must be saved to disk again, with a different, unique filename. One solution is to use a new name every time you save the program, perhaps using numbers to keep the versions straight. Under this scheme, you might have programs named SPRITE1, SPRITE2, and so on. This wastes disk space and also requires that you remember which number to use next. Later, you must scratch all the old versions and rename the most recent version with the final name.

Another option is to use a save-with-replace command which, unless you are very careful, may damage an entire disk. To avoid this danger completely, you can scratch the original program before saving a new copy. This may be time consuming, particularly if you are the careful type who likes to validate the disk or verify the saved program afterward.

"Resave" is a short, convenient utility designed to simplify the process of resaving programs. Type in the program and save it. To install Resave in memory, simply load the program and run it; then type NEW and press RETURN. Now load the program you wish to edit.

When you are ready to resave the program, type SYS 49152 and press RETURN to activate Resave. You will be prompted for a filename. Enter the name of the program to resave. Do not include a 0: before the filename, since this utility automatically adds the drive prefix to the name. Resave scratches the original copy of the program, validates the disk to recover unused disk space, saves the new edition of the program, and verifies the save.

Note that Resave will not work with machine language programs unless they are designed to load and run like BASIC programs. Since Resave occupies memory beginning at location 49152, it cannot be used with any other program or utility that uses the same space.

You might prefer to eliminate the disk validation from Resave, since validation can take a long time for a disk that contains many files. To make this alteration, change the value 86 in line 100 to 73.

Resave

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

```
FC 1 REM *COPYRIGHT 1987*
GF 2 REM *COMPUTE! PUBLICATION
   S, INC.**
SH 3 REM *ALL RIGHTS RESERVED*
CA 4 PRINT"{CLR}{12 SPACES}COP
   YRIGHT 1987":PRINT"
   {6 SPACES}COMPUTE! PUBLIC
   ATIONS, INC."
GS 5 PRINT"{9 SPACES}ALL RIGHT
   S RESERVED"
RP 6 FOR X=1TO1500:NEXT
QQ 10 FOR T=49152 TO 49360:REA
   D S:POKET,S:NEXT:PRINT"
```

```
{CLR}{DOWN}SYS 49152 TO
{SPACE}UTILIZE"
CM 20 DATA76,55,192,70,73,76
JE 30 DATA69,78,65,77,69,32
BD 40 DATA40,60,61,49,54,32
QG 50 DATA67,72,65,82,65,67
AX 60 DATA84,69,82,83,41,63
DJ 70 DATA32,0,32,32,32,32
DH 80 DATA32,32,32,32,32,32
GG 90 DATA32,32,32,32,32,32
PK 100 DATA32,32,32,0,86,48
JM 110 DATA0,169,3,160,192,32
GB 120 DATA30,171,162,0,32,207
HH 130 DATA255,201,13,240,7,15
   7
SD 140 DATA36,192,232,76,64,19
   2
FE 150 DATA232,134,2,169,1,162
MK 160 DATA8,160,15,32,186,255
JH 170 DATA169,83,141,33,192,1
   69
QF 180 DATA48,141,34,192,169,5
   8
BQ 190 DATA141,35,192,166,2,23
   2
XG 200 DATA232,138,133,2,162,3
   3
CK 210 DATA160,192,32,189,255,
   32
HD 220 DATA192,255,32,195,255,
   169
CB 230 DATA1,162,8,160,15,32
KB 240 DATA186,255,169,2,162,5
   2
CM 250 DATA160,192,32,189,255,
   32
HQ 260 DATA192,255,32,195,255,
   169
DE 270 DATA1,162,8,160,1,32
XX 280 DATA186,255,166,2,202,1
   38
RB 290 DATA162,34,160,192,32,1
   89
JQ 300 DATA255,169,43,166,45,1
   64
RH 310 DATA46,32,216,255,169,1
DM 320 DATA162,8,160,1,32,186
MQ 330 DATA255,166,2,202,138,1
   62
DS 340 DATA34,160,192,32,189,2
   55
HP 350 DATA169,1,32,213,255,16
   9
EM 360 DATA1,32,195,255,96,0
```

©

Full-Screen Editor For Applesoft

Alex Wong

This convenient, full-featured editor makes it much easier to edit Applesoft BASIC programs. The program runs only in DOS 3.3.

One of the least popular features of Applesoft BASIC is its primitive means of editing a program line. "Full-Screen Editor" for the Apple II series gives you the benefit of a full-screen editor within Applesoft BASIC. With this program, you can list the part of the program you wish to change, move the cursor to the desired spot, change the line as needed, and enter the new line in memory.

This program works on any Apple II-series computer using DOS 3.3. Enter the program with the "MLX" machine language entry program printed elsewhere in this issue. When you run MLX, you'll be asked for a starting address and an ending address for the data you'll be entering. Here are the values to use for Full-Screen Editor:

STARTING ADDRESS? 8F00
ENDING ADDRESS? 95FF

Be sure to save a copy of the program before trying to run it. You can start Full-Screen Editor with a BRUN command. For instance, if

you saved the program with the name FSE, type BRUN FSE and press Return to load and activate the editing features. The program lowers HIMEM to protect itself.

Listing The Lines To Edit

Once the editor has been installed, you can load the BASIC program you wish to edit. To prepare for editing, type a slash (/) followed by a range of line numbers that indicate which lines you want to display on the screen. This range can have the following forms:

100	list line 100
100-	list lines from 100 to the end of the program, or until the Esc key is pressed
100-200	list lines 100 to 200
-100	list from the beginning of the program to line 100

The Full-Screen Editor displays BASIC lines somewhat differently from the way Applesoft BASIC displays them. All line numbers start in column 1 with the first statement of the line starting in column 5. Lines too long to fit across one screen line will start in column 2 of the next line. There is only one space between keywords, and all control characters are displayed in inverse mode. The bottom of the screen will show CURSOR MODE

or EDIT MODE, depending on the mode you are currently in.

When you invoke the editor, it always clears the bottom of the screen with a listing of the lines you specified. While lines are being listed, the space bar can be used to pause and restart the listing. If you invoke the editor without specifying a range of line numbers, it simply clears the bottom of the screen.

Cursor Mode

Once you have listed the desired lines on the screen, the Full-Screen Editor goes into cursor mode. In this mode you can move the cursor anywhere on the screen or enter edit mode to change a line. Use the I, J, K, M, and the left and right arrow keys to position the cursor on the spot where you want to make a change. Once you have the cursor positioned where you want it, press Control-E (hold down Control and press E) to enter edit mode. The Esc key exits the editor and returns you to BASIC.

Edit Mode

Edit mode allows you to change any line displayed on the screen. Most editing commands are performed by pressing Control together with a letter key. Following is a

list of the editing commands.

Control-J Cursor down
Control-K Cursor up
left arrow Cursor left
right arrow Cursor right
Control-I Insert a blank space under the cursor and move following characters right
Control-D Delete the character under the cursor and move following characters left
Return Enter the line into memory
Esc Exit edit mode, returning to cursor mode

You can press Return anywhere on a line to enter it into memory. The beginning of a line is marked by its line number, so be sure that all line numbers begin in the first column of the screen.

Full-Screen Editor

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

```

8F00: 20 2F FB 20 AE 94 20 58 DC
8F08: FC A0 BF A9 16 20 48 95 B1
8F10: 20 3F 94 4C BC 95 06 13 40
8F18: 05 20 16 31 2E 30 13 20 10
8F20: 20 01 17 05 20 31 39 38 33
8F28: 36 8D 00 20 1B FD C9 8D B9
8F30: D0 15 48 E6 76 F0 06 A9 A1
8F38: DD C5 33 D0 07 A9 AF CD 37
8F40: 00 02 F0 04 68 C6 76 60 E9
8F48: 68 9D 00 02 20 39 D5 86 3B
8F50: 76 86 B8 84 B9 A2 13 9D C7
8F58: E8 95 CA 10 FA 85 24 20 02
8F60: 3F 94 A5 00 BD F6 95 A5 12
8F68: 01 8D F7 95 A9 0F 8D E5 4E
8F70: 93 20 2F FB A2 17 86 23 A8
8F78: A9 11 85 25 20 42 FC 20 D7
8F80: AE 94 A2 01 20 34 95 20 9D
8F88: B4 94 20 B1 00 20 B1 00 2A
8F90: 20 6E 92 4E E8 95 A2 0F 84
8F98: 8E E5 93 A2 01 20 34 95 9B
8FA0: 20 0C FD C9 9B F0 2D C9 F3
8FAB: 85 F0 0D C9 E0 90 02 29 7B
8FB0: DF 20 0D 90 B0 E0 90 E8 85
8FB8: A2 02 20 34 95 20 0C FD 34
8FC0: C9 A0 90 06 20 ED FD 4C 60
8FC8: BD 8F C9 9B F0 C5 20 03 7F
8FD0: 90 4C B8 8F AD F6 95 85 55
8FD8: 00 AD F7 95 85 01 A2 18 49
8FE0: 86 23 CA 86 25 20 11 92 2C
8FE8: 20 42 FC A9 2B 85 38 A9 6C
8FF0: 8F 85 39 20 93 FE 20 EA 26
8FF8: 03 A9 FF 85 76 A2 EF 9A 15
9000: 4C D0 03 A2 95 A0 AB 84 11
9008: 9D A0 0C D0 08 A2 95 A0 45
9010: 9C 84 9D A0 0C 20 26 90 1C
9018: A2 95 A0 BA 84 9D A0 06 91
9020: 20 26 90 4C 68 94 86 9E 0B
9028: D1 9D F0 06 88 88 10 9F
9030: F7 60 68 68 C8 B1 9D 85 C6
9038: 3C C8 B1 9D 85 3D 6C 3C EF
9040: 00 20 C1 93 A4 24 D0 07 39
9048: B1 28 20 44 92 90 3E C8 B0
9050: C4 21 F0 0B B1 28 88 91 BB
9058: 28 C8 C8 C4 21 D0 F5 88 E5
9060: A9 A0 91 28 E6 25 A5 25 6F
9068: C5 23 B0 1F A5 28 85 2A 40
9070: A5 29 85 28 20 22 FC A0 36
9078: 00 B1 28 20 44 92 B0 0D E7
9080: C8 B1 28 A0 27 91 2A A0 F5
9088: 02 90 C9 C6 25 4C 9B 93 99
9090: 20 C1 93 A0 27 B1 28 8D 8C
9098: EE 95 88 B1 28 C8 91 28 72
90A0: 88 88 C4 24 10 F5 A4 24 C8
90AB: A9 A0 91 28 A0 00 84 24 AD
90B0: E6 25 A5 25 C5 23 D0 2B 1D

```

```

90B8: C6 25 AD EE 95 C9 A0 F0 31
90C0: 1F 20 E6 93 A5 25 48 20 02
90C8: 9B 93 A5 25 C5 22 F0 02 3E
90D0: C6 25 20 C1 93 68 20 C1 FE
90D8: FB A0 01 AD EE 95 91 28 34
90E0: 4C 9B 93 20 C1 FB A0 00 C2
90E8: B1 28 C9 A0 F0 0C AD EE 32
90F0: 95 C9 A0 F0 EB A6 25 20 D6
90F8: 00 94 A0 00 AD EE 95 91 39
9100: 28 4C 93 90 60 A6 25 8E 3C
9108: FB 95 20 4F 92 90 F5 8E D8
9110: EC 95 86 25 20 22 FC A2 58
9118: 00 8E F0 95 B1 28 C9 A0 B8
9120: B0 02 29 3F 29 7F 9D 00 B7
9128: 02 E8 C8 C4 21 90 ED EE 02
9130: F0 95 E6 25 A5 25 C5 23 D0
9138: D0 02 B0 18 20 22 FC A0 FF
9140: 00 B1 28 C9 A0 D0 0D AD 81
9148: F0 95 C9 06 F0 03 C8 D0 D8
9150: CB EE F0 95 2C EC 95 30 FC
9158: EC C6 25 20 22 FC CA A0 2E
9160: 27 B1 28 C9 A0 D0 10 CA 58
9168: 88 10 F6 E8 CE F5 95 C6 6D
9170: 25 20 22 FC 4C 5F 91 8E D3
9178: F5 95 20 11 92 AD F0 95 D3
9180: C9 07 D0 14 84 24 A2 03 A2
9188: 20 34 95 20 0C FD C9 8D F6
9190: D0 05 4E E8 95 10 11 60 25
9198: E6 25 A5 25 C5 23 D0 08 E3
91A0: C6 25 8E F5 95 20 E6 93 30
91A8: AE F5 95 BD 00 02 C9 0D D7
91B0: F0 01 E8 A9 00 9D 00 02 BC
91B8: 20 BD 94 20 C1 93 AD FF 91
91C0: A0 01 4C 44 D4 20 1B 92 32
91C8: 4E EA 95 A5 24 D0 10 20 7F
91D0: 16 92 A5 25 C5 22 F0 5C 9F
91D8: C6 25 20 22 FC 18 60 C6 9E
91E0: 24 4E EF 95 18 60 38 6E 22
91E8: EF 95 20 1B 92 E6 24 A5 3D
91F0: 24 C5 21 90 EC 20 11 92 61
91F8: 2C EC 95 30 E4 38 6E F8 01
9200: 95 E6 25 A5 25 C5 23 B0 DF
9208: 03 4C C1 FB C6 25 4C E6 04
9210: 93 A9 00 85 24 60 A9 27 DE
9218: 85 24 60 2C FB 95 10 14 42
9220: A4 24 B1 28 29 DF C9 C1 77
9228: 90 0A C9 DB B0 06 B1 28 38
9230: 49 20 91 28 60 20 84 94 D7
9238: 20 BD 94 20 B7 00 20 C7 7B
9240: DA 4C 99 94 C9 BA B0 05 02
9248: C9 B0 90 01 60 18 60 A5 6A
9250: 25 85 35 A0 00 8A 20 C1 46
9258: FB B1 28 20 44 92 B0 04 C0
9260: CA 10 F2 18 08 A5 35 20 30
9268: C1 FB 28 18 60 20 B7 B2
9270: 00 F0 F9 20 35 92 20 1A 61
9278: D6 BA 8E FA 95 20 B7 00 D5
9280: F0 14 C9 FF D0 08 A9 FF 56
9288: 85 50 85 51 D0 08 20 B1 E2
9290: 00 F0 F3 20 35 92 A0 01 A8
9298: B1 9B F0 78 C8 B1 9B A4 12
92A0: C8 B1 9B C5 51 D0 04 E4 21
92AB: 50 F0 02 B0 67 84 85 2C 0E
92B0: F8 95 10 05 2C F9 95 30 AE
92B8: 5B 2E E8 95 30 03 20 52 8D
92C0: 94 A0 00 8C EB 95 84 24 04
92C8: 20 24 ED A9 A0 A6 24 E0 28
92D0: 05 F0 06 20 72 93 E8 D0 FB
92D8: F6 A4 85 C8 B1 9B 30 0E 4A
92E0: A9 A0 20 72 93 D0 03 20 34
92E8: 72 93 E6 85 A4 85 B1 9B 9B
92F0: D0 26 A8 A9 0D 20 72 93 19
92F8: AD EA 95 D0 E9 95 30 08 A1
9300: A9 A0 20 72 93 4C FB 92 41
9308: B1 9B AA C8 B1 9B 86 9B 75
9310: 85 9C D0 82 4E EA 95 60 0D
9318: 10 CD AE 00 C0 E0 A0 D0 2C
9320: 0B 2C 10 C0 AE 00 C0 10 EC
9328: FB 2C 10 C0 E0 9B F0 52 10
9330: 38 E9 7F AA 84 85 A0 D0 D4
9338: 84 9D A0 CF 84 9E 20 4E 47
9340: 93 A4 85 C8 B1 9B F0 A6 1C
9348: 30 A4 A9 20 D0 99 A0 FF 16
9350: CA F0 07 20 2C D7 10 FB D8
9358: 30 F6 A5 9E C9 CF 90 05 A7

```

```

9360: A9 20 20 72 93 20 2C D7 DC
9368: 30 05 20 72 93 D0 F6 4C 2E
9370: 72 93 48 09 80 20 ED FD AD
9378: 2C F8 95 10 0D 2C F9 95 4A
9380: 10 08 AE FA 95 9A 2C 10 B6
9388: C0 60 68 60 48 A5 25 38 96
9390: 69 00 C5 23 D0 03 6E F9 C0
9398: 95 68 60 08 8E FF 02 BA 64
93A0: 8E FE 02 AE E5 93 9A 68 15
93A8: 85 25 68 85 24 68 8D F9 19
93B0: 95 BA 8E E5 93 AE FE 02 D8
93B8: 9A 20 22 FC AE FF 02 28 EA
93C0: 60 8E FF 02 BA 8E FE 02 EB
93C8: AE E5 93 9A AD F9 95 48 A5
93D0: A5 24 48 A5 25 48 BA 8E 85
93D8: E5 93 AE FE 02 9A AE FF 75
93E0: 02 4E F9 95 60 0F 48 98 9D
93E8: 48 2C EC 95 30 08 A0 00 19
93F0: 8C EA 95 BC F9 95 20 70 6B
93F8: FC 20 22 FC 68 AB 68 60 D1
9400: A5 22 48 86 22 A5 23 38 1C
9408: E9 01 48 20 24 FC A5 28 F9
9410: 85 2A A5 29 85 2B A4 21 11
9418: 88 68 C5 22 F0 0F E9 01 13
9420: 48 20 24 FC B1 28 91 2A 45
9428: 88 10 F9 30 E1 A5 22 20 E5
9430: C1 FB A0 27 A9 A0 91 28 DA
9438: 88 10 FB 68 85 22 60 A9 CE
9440: 8F A2 2B C5 74 90 60 D0 5E
9448: 08 E4 73 B0 04 85 74 86 CD
9450: 73 60 48 20 11 92 8D EA 2F
9458: 95 E6 25 A5 25 C5 23 D0 5C
9460: 05 C6 25 20 E6 93 20 22 4C
9468: FC 68 60 A0 20 8C FF 02 75
9470: AD 30 C0 CC FF 02 D0 03 0E
9478: AD 30 C0 88 D0 F5 CE FF 21
9480: 02 10 ED 60 A5 36 8D F1 85
9488: 95 A5 37 8D F2 95 A5 38 17
9490: 8D F3 95 A5 39 8D F4 95 0A
9498: 60 AD F1 95 85 36 AD F2 48
94A0: 95 85 37 AD F3 95 85 38 F0
94AB: AD F4 95 85 39 60 20 89 06
94B0: FE 4C 93 FE A9 C6 85 36 78
94B8: A9 94 85 37 60 A9 FE 85 2D
94C0: 36 A9 94 85 37 60 48 C9 EF
94C8: A0 B0 02 29 3F 48 84 35 9A
94D0: 2C EA 95 10 0B 20 52 94 90
94D8: A9 A0 A4 24 91 28 E6 24 F4
94E0: A4 24 68 91 28 E6 24 A5 56
94E8: 24 C5 21 D0 08 6E EA 95 2C
94F0: C6 24 20 8C 93 A5 F1 20 8A
94F8: A8 FC 68 A4 35 60 C9 A0 6C
9500: 90 FB C9 DD F0 14 20 4A EB
9508: FF AD ED 95 D0 03 20 78 01
9510: 95 20 3F FF EE ED 95 4C 9C
9518: F6 FD AD ED 95 F0 09 20 75
9520: 6B 94 20 0C FD 4E E8 95 7B
9528: 20 78 93 20 B4 94 A2 EF EB
9530: 9A 4C 96 8F 20 3A 95 4C E8
9538: 9B 93 A9 C3 85 9D A9 94 12
9540: 85 9E 20 78 85 9C 4E 93 6F
9548: 85 9D 84 9E 20 58 FC A0 15
9550: 00 B1 9D F0 20 C9 81 90 66
9558: 11 C9 8D B0 0D 29 0F AA 11
9560: A9 A0 20 F6 FD CA D0 FA B3
9568: F0 03 20 F6 FD E6 9D D0 D7
9570: E0 E6 9E D0 DC 4C 0C FD D4
9578: 20 C1 93 A9 17 20 C1 FB E9
9580: 20 11 92 8D EA 95 8D ED E1
9588: 95 4C 9C FC A9 EB 8D FD FF
9590: 03 A9 8F 8D F3 03 20 6F CD
9598: FB 4C EB 8F 95 EA 91 C9 90
95A0: D2 91 CA C8 91 CB ED 91 A8
95AB: CD FD 91 8D 05 91 FF 41 F4
95B0: 90 84 41 90 89 90 90 95 BB
95B8: E6 91 BA FD 91 8B D2 91 DE
95C0: 88 C5 91 43 55 52 53 4F F1
95C8: 52 20 4D 4F 44 C5 45 44 CB
95D0: 49 54 20 4D 4F 44 C5 56 FB
95D8: 45 52 49 46 59 3F 20 28 FB
95E0: 45 53 43 2F 47 45 54 A9 80
95E8: 00 00 00 00 00 38 46 39 BA
95F0: 35 46 30 39 35 46 30 35 3A
95F8: 46 38 46 38 38 33 41 00 B2

```



RAMtest 128

Jim Butterfield, Associate Editor

Have you ever suspected that your Commodore 128 might have a bad memory chip? This program can provide an accurate answer.

The 128 has a lot of RAM (Random Access Memory). Just one bad byte could make your programs misbehave. How can you be sure that your 128's memory is OK? "RAMtest 128" performs a thorough memory test.

Programmers often suspect hardware problems when their newest programs fail to behave correctly. They are usually wrong; it's almost always a programming error, not a problem with the machine. But on rare occasions memory does fail, and it's useful to have a program that can confirm or deny such a suspected problem.

Using The Program

RAMtest 128 is written in machine language (ML). The program can be entered using the "MLX" machine language entry utility published regularly in COMPUTE!. If you don't have a copy of 128 MLX (found in

the April issue), you can use the 64 version (in 64 mode) of MLX, found elsewhere in this issue. MLX will request a starting address and an ending address for the data you'll be entering. For RAMtest 128, use these values:

Starting address: 1C01
Ending address: 1CD1

Be sure you save a copy of the RAMtest 128 data before leaving MLX.

You can also bypass MLX and use the 128's built-in ML monitor to enter the data. (If you're using the 80-column display, you should first switch to the 40-column screen.) To enter the monitor, type MONITOR and press RETURN, or simply press the F8 function key. You must remember several things when entering the data using the monitor: Precede each line with a greater-than (>) symbol, substitute a space for the colon (:), following the four-digit address value, and omit the final (rightmost) two-digit value in each line. Each line you enter should consist of a > character, a four-digit address, and eight two-

digit data values. After you have entered and checked all of the lines, save the program with a command of the form:

```
S "filename",08,1C01,1CD0
```

(For tape, substitute 01 for 08 in the command above.) After saving the program, type X and press RETURN to exit the monitor and return to BASIC.

When you're ready to conduct the test, load the RAMtest 128 program as you would any BASIC program. Use RUN to start the test. Every 20 seconds the program reports OK and does the test again. If you're using the 80-column display, type FAST before you run the program to double its speed. Press the reset button (near the power switch) to stop the program.

About Memory Tests

It seems simple to devise a memory test. Just store something in every memory cell you can conveniently reach, and then read it back to insure that the data was stored correctly. But that's not good enough. A good test should check for correct

addressing, as well.

Let's outline the kind of problem that simple tests won't catch. Whenever you do something that requires access to a memory address, the computer's processor signals the address it wants to use by sending it over the *address bus*—a group of 16 wires that together define the address. Now suppose there's something wrong with the address bus or its connections, so that a memory chip doesn't read all the wires correctly. The chip still sees an address, but it's the wrong one. If this happens, the data is stored in the wrong address. Worse, when the processor reads the data back to check it, it reads from the same wrong address. The data seems to be right, but it's not.

Here's a specific example. Suppose that you try to store data to address 5000, but an error on the address bus causes the address to be seen as 4992. (Hardware hackers will spot this as an error in line 3 of the address bus.) The processor stores something there, thinking that the data is going to 5000. But it really goes to 4992. Now the processor tries to test memory by recalling the contents of 5000. It sends out the address again. The same error causes the address to be seen as 4992, and the contents of 4992 are delivered to the processor. The processor sees that it's the same value that was stored and does not report a memory error.

A Better Algorithm

Several good methods for testing memory have been developed. RAMtest 128 uses a shortened version of a method published by Knaizuk and Hartmann in 1977. In this scheme, individual memory cells are tested in the usual way, by storing a value and reading it back. A special wrinkle is used to detect addressing faults.

A single fault in the addressing system causes an address error that is a power of 2. In our example above, the error was 8, which is 2 raised to the third power. Depending on the address line that is at fault, the error might be 2, 4, 8, 16, 32, 64, 128, or other power of 2 up to 32768. Notice that none of these numbers may be divided evenly by 3.

To detect addressing faults, we

store to memory in a pattern of three bytes: the first byte with one value, and the next two with a different value. Any power-of-2 error will disrupt the three-byte pattern. When the program sweeps through memory checking for correct values, it will find a misplaced value and know there's a problem.

Inside The Test

The testing portion of this program stores a value (hex \$55, which has the binary pattern %01010101) in every location it can reach in both bank 0 (\$0B00-\$FEFF) and bank 1 (\$0400-\$FEFF). Then it goes back and puts a different value (hex \$AA, which has the binary pattern %10101010) into every third location. It's important, by the way, that this be done as a separate program step. Finally, the program sweeps through both banks, checking that all locations contain the expected values.

After the first sweep, the program reverses the test values and does it all over again. If everything checks out, RAMtest 128 prints OK on the screen and repeats the test. If an error occurs, the program transfers control to the 128's built-in machine language monitor.

Why does the test repeat? Some memory failures are intermittent; they might be brought on by heat, power supply variations, or even aging components. If that's your suspicion, let it run; you'll see a lot of OK messages printed.

If the program stops with an error, you'll end up in the machine language monitor. If you're familiar with machine language, the contents of memory locations \$FA-\$FC will give you a hint as to the location of the error. If you're not, type M FA FA and copy down the computer's response in order to show it to someone who understands the hardware side.

Program Setup

This program is constructed to take into account some oddities of the 128's configuration. If you have executed a GRAPHIC command in the current session, BASIC might start at an address that's higher than normal. Thus, the program begins with the command BANK 0. Without this command, we might

SYS to ROM addresses instead of to our program in RAM. The SYS command doesn't go to an absolute address, but to an offset relative to the beginning of BASIC. Before it begins the test, the program relocates the testing section to the memory area at location \$1000. The test cleans away the relocating code as well as the original copy of the test code.

Scope

The program tests over 123K of memory (to be exact, 126,720 bytes) in 20 seconds—even less if you're in FAST mode. That's over 6000 bytes per second, which is very fast indeed. Each byte is written at least twice and then checked twice in a mere 160 microseconds or so.

A few areas are not tested. The program doesn't go below address 2816 (\$0B00) in bank 0, since that would garble the operating system information and the screen display. And it doesn't go above 65279 (\$FEFF) to avoid disturbing the MMU (Memory Management Unit) registers and the sensitive interrupt routines which reside there.

It will be extremely rare to find memory problems; there are many more bugs in software than in hardware. If you think you've found a problem, check the program on a friend's computer before you take your machine in for service.

RAMtest 128

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

```
1C01:1D 1C 64 00 FE 02 30 3A F5
1C09:9E C2 28 34 35 29 AA C2 EF
1C11:28 34 36 29 AC 32 35 36 92
1C19:AA 33 30 00 00 00 A0 2E E8
1C21:B1 2D 99 D2 00 C8 C0 CE 51
1C29:D0 F6 78 4C 00 01 20 6F 0F
1C31:01 AD 9E 01 91 FA 20 7F 71
1C39:01 90 F9 20 6F 01 AD 9F D1
1C41:01 91 FA 20 7F 01 B0 0A 2B
1C49:20 7F 01 B0 05 20 7F 01 45
1C51:90 EF 20 6F 01 B1 FA CD 5B
1C59:9F 01 D0 2C 20 7F 01 B0 30
1C61:18 B1 FA CD 9E 01 D0 20 09
1C69:20 7F 01 B0 0C B1 FA CD A7
1C71:9E 01 D0 14 20 7F 01 90 26
1C79:DC AD 9E 01 AE 9F 01 8D F2
1C81:9F 01 8E 9E 01 10 A7 A9 C6
1C89:00 8D 00 FF A9 4F 20 D2 C2
1C91:FF A9 4B 20 D2 FF A9 20 A9
1C99:20 D2 FF D0 91 A0 3F 8C BD
1CA1:00 FF A0 0B 84 FB A0 00 F3
1CA9:84 FA 84 FC 60 18 C8 D0 09
1CB1:FB E6 FB A6 FB E0 FF D0 BF
1CB9:F3 A2 04 86 FB AE 00 FF 18
1CC1:E0 7F B0 E8 E6 FC A2 7F DE
1CC9:8E 00 FF 60 AA 55 00 00 F9
```

©

ML Runner

Paul Lindner

This Commodore 64 utility makes machine language programs easier to handle by turning them into files that load and run like BASIC. No machine language expertise is needed to use the program.

Some of the best programs available for the Commodore 64 are written in machine language—the computer's native tongue. But machine language files aren't as easy to use as BASIC programs. Most require special loading procedures, and it's easy to forget the SYS address that starts up a program. But it doesn't have to be that way. COMPUTE!'s *SpeedScript* word processor, for instance, loads and runs exactly like BASIC, even though it's written entirely in machine language. Wouldn't it be convenient if every machine language program worked that way?

"ML Runner" does exactly that, converting a machine language program into a program file that saves, loads, and runs as if it were a BASIC program. Type in the program and save a copy on disk or tape. Then select the program you wish to convert. In order to do this, you must know four facts about the machine language program: its name, starting address, ending address, and the SYS address normally used to run the program.

After you run the program, respond to the prompts as indicated. Be sure to use a different name for the new file you are about to create. When typing the starting and ending addresses, you can use either decimal or hexadecimal numbers. Hexadecimal numbers must be preceded with a dollar sign (\$). These addresses are the same ones printed in the article for any COMPUTE! program which is typed in with the "MLX" machine language entry program.

After you answer all the questions, the program goes to work, displaying the number of bytes it has already processed. When it is done, the file is ready to use. Load it and type RUN, as you would any BASIC program. The converted program is also much easier to copy than the original. To make a new copy of the program, simply load it into memory and then save a new copy as if this were a normal BASIC program.

This method can be used even if you don't know all the required address information. This short program will display the starting address of any program on disk:

```
QG 10 OPEN 2,8,2,"0:FILENAME,P
,R"
PA 20 GET#2,A$,B$:CLOSE 2
BH 30 PRINT ASC(A$+CHR$(0))+25
6*ASC(B$+CHR$(0))
```

Finding the ending address of a program is even easier. Enter these

two commands in direct mode (without line numbers):

```
LOAD"FILENAME",8,1
PRINT PEEK(46)*256+PEEK(45)
```

In both of these examples, you should replace FILENAME with the name of the program you wish to investigate. The SYS address of a machine language program is usually the same as the starting address.

There are a few programs for which this technique is unsuitable. Obviously, if a program already loads and runs like BASIC, there is no point in converting it with this program. Some programs, including much commercial software, take over the computer when you load them with ,8,1 (they start without requiring that you enter a SYS command). Programs of this type rely on special features of the computer and may not work at all if you convert them with this program.

ML Runner

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

```
RH 10 REM *COPYRIGHT 1987*
FQ 20 REM *COMPUTE! PUBLICATIO
NS, INC.*
KP 30 REM *ALL RIGHTS RESERVED
*
QK 40 DEF FN HI(Q)=INT(Q/256)
GE 50 DEF FN LO(Q)=Q-256*FN HI
(Q)
FJ 60 PRINT"{CLR}COPYRIGHT 198
7 COMPUTE! PUBLICATIONS"
BD 70 PRINT"{8 SPACES}ALL RIGH
TS RESERVED"
KX 80 FOR X=1TO1500:NEXT
```