

SoftSide

April 1982 Vol. 5 No. 7 THREE DOLLARS

Special Edition



Nancy LeSaint

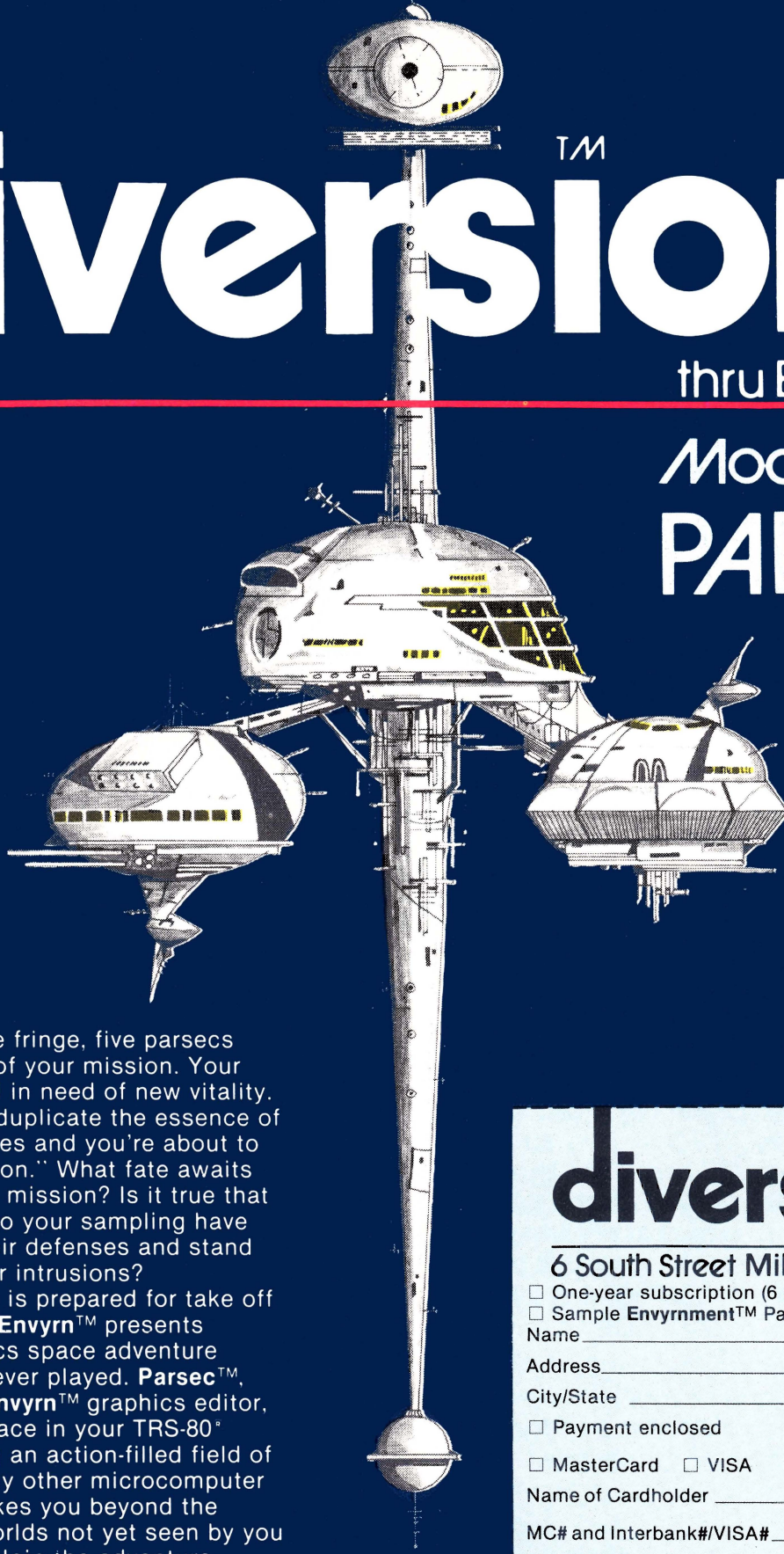
Word Processing

diversionsTM

thru Envyrn

Module One

PARSECTM



You hover on the fringe, five parsecs from initialization of your mission. Your culture is tired and in need of new vitality. The Chamber can duplicate the essence of life from any species and you're about to start your "collection." What fate awaits you on this unique mission? Is it true that cultures opposed to your sampling have been preparing their defenses and stand ready to resist your intrusions?

The first module is prepared for take off — **diversions thru EnvyrnTM** presents **ParsecTM**, a graphics space adventure unlike any you've ever played. **ParsecTM**, created with the **EnvyrnTM** graphics editor, simulates deep space in your TRS-80[®] computer, creating an action-filled field of play larger than any other microcomputer game. **ParsecTM** takes you beyond the screen and into worlds not yet seen by you or your computer. Join the adventure — subscribe to **diversions thru EnvyrnTM** now or send \$20 for **ParsecTM**, the first of six modules to be released in 1982. Each module includes a magazine and a disk.

diversionsTM

thru Envyrn

6 South Street Milford, NH 03055

- One-year subscription (6 modules) — \$60.00
 Sample **EnvyrnTM** **ParsecTM** only — \$20.00

Name _____

Address _____

City/State _____ Zip _____

Payment enclosed

MasterCard VISA Exp. Date _____

Name of Cardholder _____

MC# and Interbank#/VISA# _____

Signature _____

To order use the card provided in this issue or send in this order blank.
I own a 48K TRS-80[®] with Disk Model I Model III

TRS-80 is a registered trademark of Tandy Corporation

HAVE WE GOT A PROGRAM FOR YOU IN '82

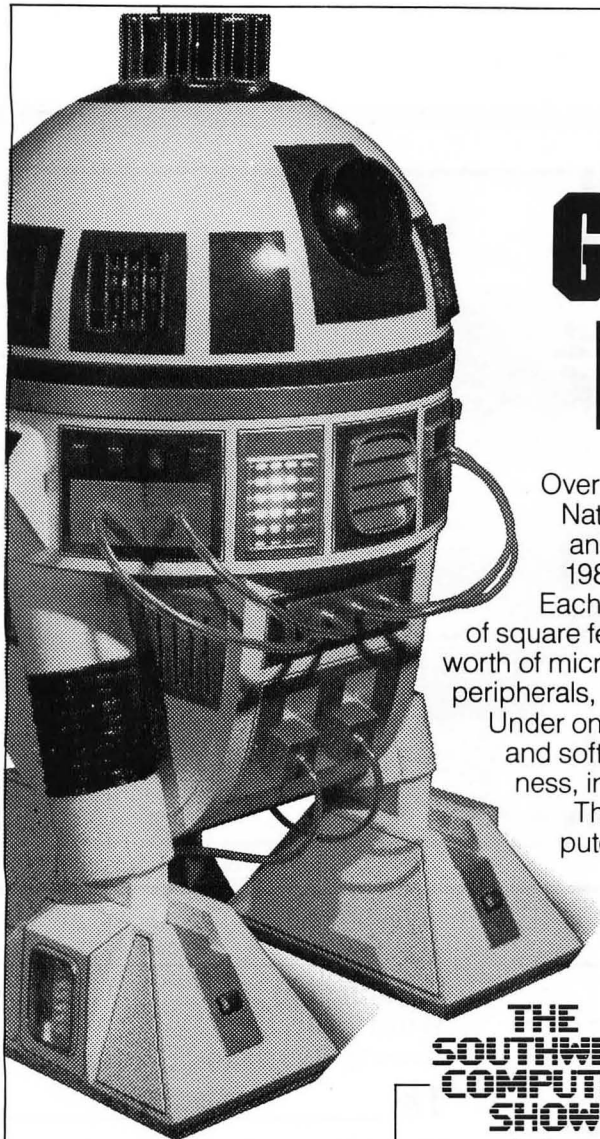
Over 150,000 computer owners and novices attended the 1981 National Computer Shows and Office Equipment Expositions, and more than a quarter of a million are expected to be at the 1982 shows.

Each show features **hundreds** of companies using **thousands** of square feet of display space to showcase and sell **millions** of dollars worth of micro and mini computers, data and word processing equipment, peripherals, accessories, supplies and software.

Under one roof you'll see — and be able to buy — all of the hardware and software made by every major computer manufacturer for business, industry, government, education, home and personal use.

The show includes computers costing as little as \$100 to computers selling for \$150,000.

Don't miss the coming of the new computers — show up for the show. Admission is \$5 per person and \$3 for children.



THE NATIONAL COMPUTER SHOWS

Ticket Information

Send \$5 with the name of the show you plan to attend to National Computer Shows, 824 Boylston Street, Chestnut Hill, Mass. 02167. Tickets can also be purchased at the show.

THE MID-ATLANTIC COMPUTER SHOW

Washington, DC
DC Armory/Starplex
Across from RFK Stadium

Thursday-Sunday
October 28-31, 1982
11 AM to 6 PM Daily

DIRECTIONS:
2001 E. CAPITOL ST. SE
(E. CAPITOL ST. EXIT OFF I-295
— KENILWORTH FRWY)

THE SOUTHWEST COMPUTER SHOW

Dallas
Dallas Market Hall

Thursday-Sunday
April 15-18, 1982
11 AM to 6 PM Daily

DIRECTIONS:
2200 STEMMONS FREEWAY
(AT INDUSTRIAL BLVD)

THE NEW YORK COMPUTER SHOW

Uniondale, Long Island
Nassau Coliseum

Thursday-Sunday
April 22-25, 1982
11 AM to 6 PM Daily

DIRECTIONS: TAKE L.I. EXPWY
TO EXIT 38 NO. STATE PKWY
TO EXIT 31A MEADOWBROOK
PKWY SO. TO EXIT M5
HEMPSTEAD TURNPIKE

THE TWIN CITIES COMPUTER SHOW

Minneapolis
Minn. Auditorium
& Convention Hall
Third Avenue

Thursday-Sunday
September 16-19, 1982
11 AM to 6 PM Daily

DIRECTIONS: HWY 94 to
11th St. Exit to Third Ave.

THE MID-WEST COMPUTER SHOW

Chicago
(Arlington Heights)
Arlington Park Racetrack
Exhibition Center

Thursday-Sunday
November 5-7, 1982
11 AM to 6 PM Daily

DIRECTIONS: EUCLID AVE &
WILKE RD. TAKE NW TOLLWAY
TO RTE 53 EXIT AT
EUCLID AVE EAST

THE NORTHEAST COMPUTER SHOW

Boston
Hynes Auditorium/
Prudential Center

Thursday-Sunday
November 11-14, 1982
11 AM to 6 PM Daily

DIRECTIONS: TAKE MASS
PIKE TO PRUDENTIAL
CENTER EXIT

THE SOUTHEAST COMPUTER SHOW

Atlanta
Atlanta Civic Center

Thursday-Sunday
December 9-12, 1982
11 AM to 6 PM Daily

DIRECTIONS:
395 PIEDMONT AVE NE
(AT RALPH MCGILL BLVD)

The National Computer Shows are produced by Northeast Expositions Inc. who also produce Electronica — shows featuring home entertainment equipment and personal electronics — which are held annually in major US cities. NEI also produces the Applefest Shows. For more information about any of these events call us at 617-739-2000 or write to the above address.

SoftSide™

PUBLISHER
Roger W. Robitaille, Sr.

ASSOCIATE PUBLISHER
Randal L. Kottwitz

MANAGING EDITOR
ART DIRECTOR
Nancy Lapointe

PROGRAMMING EDITOR
Jon Voskuil

EDITORIAL DEPARTMENT
Scott Adams
Rich Bouchard
Mary Locke
Lance Micklus
Mark Pelczarski
Joan Truckenbrod
Alan J. Zett

ADVERTISING MANAGER
Sue Rowland

PRODUCTION MANAGER
Lynn Wood

PRODUCTION DEPARTMENT
Lynda Fedas
Karen Lawrence
Denise Lafleur

CUSTOMER SERVICE
Cindy Schalk

DEALER SALES
Kathie Maloof

STAFF
Jerry Butler
Mary Edwards
Donna Jean
Bea Kimball
Doris Miller

Photographs by Dean F. H. Macy

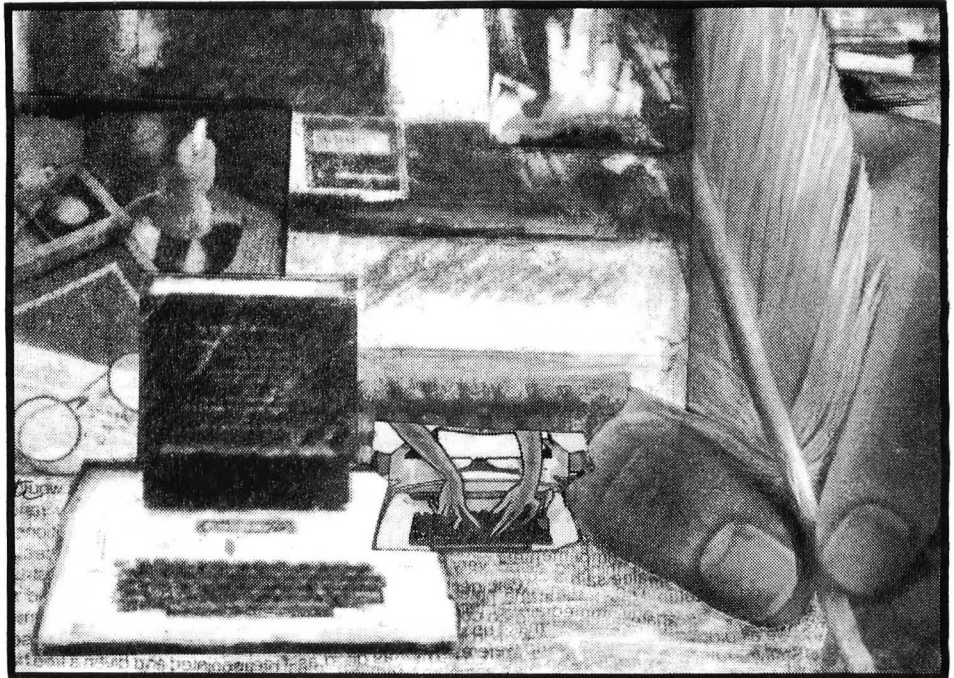
SoftSide is published each month by *SoftSide* Publications, 6 South Street, Milford, New Hampshire 03055. Telephone 603-673-0585. Second class postage paid Milford, New Hampshire and pending at additional mailing offices. ISSN: 0274-8630. Subscription rates: USA and Canada, \$30.00/12 issues. First Class USA, APO, FPO, Mexico, \$40.00/12 issues. Other foreign countries, \$62.00/12 issues. Media subscription rates: For USA, magazine and cassette, \$75.00/12 months. Magazine and disk, \$125.00/12 months. APO, FPO, Canada, Mexico, add \$20.00/12 months. Other foreign add \$50.00/12 months.. All remittances must be in U.S. funds. Mail subscription inquiries to *SoftSide* Publications, 515 Abbot Dr., Broomall, PA 19008. Entire contents © copyright March, 1982, *SoftSide* Publications. All rights reserved.

POSTMASTER - send address changes to:

SoftSide Publications
515 Abbot Drive
Broomall, PA 19008

If you do not receive your May issue of *SoftSide* by May 6, contact *SoftSide* Publications, 515 Abbot Drive, Broomall, PA, 19008 or call 1-800-345-8112 (In PA call 1-800-662-2444).

TRS-80®, Apple, and ATARI® are registered trademarks of The Tandy Corporation, The Apple Computer Company, and Warner Communications, respectively. Envyrn, Envyrrment, Envyrnese, and diversions thru Envyrn are registered trademarks of Roger W. Robitaille, Sr.



Cover illustration by Nancy Lapointe

FRONT RUNNER

21

Microtext 1.2

by Jon R. Voskuil
SoftSide's basic BASIC word processor for all three systems acquires line editing capabilities, plus a few changes to the two previous installments.

FEATURES

9

Sensuous Programmer

by "J"
"J" concludes a year-long exploration of the ins and outs of the intimate art of BASIC programming with some philosophical meanderings.

11

My Side of the Page

by Lance Micklus
For those who like to encourage their computers to talk to other people's computers, this extensive discussion of modems should prove to be very helpful.

15

Introduction to Word Processing

Review by Dave Albert

18

Word Processing: An Art in Transition

by Tom Stanton
This informative essay on the history of the written, printed, and electronic word leads to some fascinating questions about the future.

96

Machine Head

by Spyder Webb

DEPARTMENTS

5 Editorial

6 Input

8 Outgoing Mail

16 Calendar

26 Bugs, Worms, and
Other Undesirables

95 New Products

APPLE™/SIDE

34 Enhanced Disk Version

SEMAPHORE by Richard A. Bryant
Semaphore is a form of "word processing" that predates computers by quite a few years. This unique Hi-Res graphics program will enlighten you about a kind of flag-waving quite unfamiliar to most of us.

36 Disk and Cassette Versions

APPLE F.O.O.L.S. by Jon R. Voskuil
DV and CV subscribers get a sneak preview of the "Apple Fully Optimized Operational Language System" which will appear in listed form in next month's magazine.

37 K-Byter

BATTLESHIP by David Bahr
The old favorite board game is revisited in this compact computerized incarnation.

Programs

38 WHITE LIGHTNING by Randy Fox

You'll have to develop quick reflexes to keep from zapping yourself out of this game. Your score soars as you intercept elusive graphics blocks, but each time you get one it's harder to maneuver.

42 POSTER MAKER by Fred J. Condo

If you have the need (or just an irresistible urge) to print out large-letter posters, this is just what your Apple needs.

48 Comparative Review:

THREE APPLE TEXT EDITORS
SuperText II, SuperScribe II, Apple Writer II reviewed by Jon R. Voskuil

ATARI™/SIDE

54 K-Byter

CRYPTO by Jerry Aamodt
This is an entertaining code-breaking game, coded into just the form for you and your ATARI® to solve together.

55 Enhanced Disk Version

RENUMBERING FOR THE ATARI® by Frank Roberts
Anyone who has written a program from scratch knows how messy the line numbering can get after all the changing and revising is done. This program lets your computer do the grubbing work of renumbering the whole thing in nice, neat increments of your choice.

Programs

56 STARBASE 13 by Mark Lewis Baldwin
This one will give you some heavy practice with your joystick as you defend yourself against attackers from all directions. It's only a matter of time....

59 ATARI® BANNER MACHINE by Alan J. Zett
This unique banner-printing program works not only with normal characters, but with any redefined character set that you care to program into your computer. The possibilities are limitless.

63 Comparative Review:
WORD PROCESSING PROGRAMS FOR THE ATARI®
Letter Perfect, Text Wizard, and Word Processor reviewed by Sheldon Leemon

TRS-80™/SIDE

67 K-Byter

MICRO ADVENTURE by Joseph Felten
Adventure veterans and beginners alike will enjoy this sub-compact interactive game.

68 Enhanced Disk Version

SCREEN PRINT by Steven Milliken
Install a Machine Language routine into memory, and you can dump the text and graphics displayed on your screen to a printer — even while another program is running.

Programs

70 GOTHIC LETTER PRINTER Ronald M. Tutone
This remarkable program creates large letters in a beautiful Gothic font on your favorite printer. You'll have no more excuses for sloppy garage sale signs.

76 DATABASE by Mark Pelczarski
Translation of the Month, with Enhancements, by Robert Jacobs.
This is a completely updated version of *SoftSide's* Developing Database, in two versions: sequential access and random access. Those who have long awaited this complete TRS-80® version will not be disappointed.

Article

85 DOSPLUS — What Does it Mean? by Alan J. Zett
The complete list of all DOSPLUS BASIC error messages — fully explained.

Reviews

86 REFWARE'S THESAURUS by Kathleen Boucher

88 MICROPROOF by Dave A. Kater

91 HEXSPELL by Dave A. Kater and David R. Long

93 NEWSSCRIPT VERSION 6.1 by Joseph Breton

Boston

Minneapolis

Houston

San Francisco



The Most Spectacular Extravaganza Ever... For Apple Users

At Applefest '82 hundreds of manufacturers, distributors and dealers will showcase the entire spectrum of Apple-compatible products including computers, components, peripherals, plug-in cards, publications, gifts, magazines, services, accessories and software for home, office and school.

Hands-on centers and multimedia presentations will demonstrate the newest applications for business, education and entertainment.

Seminars and workshops, conducted by the world's leading Apple authorities, will detail new uses to make your Apple more enjoyable and more useful than you ever imagined.

You'll meet thousands of other Apple owners and find the newest of everything for your Apple under one roof... and for sale at super show prices.

So if you use an Apple... or are thinking about buying one, you won't want to miss a minute of Applefest '82.

Ticket & Hotel Information

Send your check and a note indicating the specific show you wish to attend. Tickets and hotel information will be mailed back to you. Tickets can also be purchased at the show. Make all checks payable to Northeast Expositions Inc. 824 Boylston Street, Chestnut Hill, Mass. 02167 Tel: 617 739 2000.

Exhibitor Information

For specific exhibitor information on one or all of the Applefest '82 shows call Northeast Expositions at the telephone number above.

Applefest/Boston

Fri-Sun May 14-16, 1982

Hynes Auditorium

Show Hours: 11AM to 6PM Daily

Admission: \$6 per day or \$10 for 2 days,
\$15 for 3 days

Applefest/Minneapolis

Thurs-Sun Sept 16-19, 1982

Minnesota Auditorium and Convention Hall

Show Hours: 11 AM to 6 PM Daily

Admission: \$5 per day or \$8 for 2 days,
\$12 for 3 days, \$15 for 4 days

Applefest/Houston

Fri-Sun Nov 19-21, 1982

Albert Thomas Convention Center

Show Hours: 1PM to 10PM Daily

Admission: \$5 per day or \$8 for 2 days, \$12 for 3 days

Applefest/San Francisco

Fri-Sun Dec 3-5, 1982

Moscone Center

Show Hours: 1PM to 10PM Daily

Admission: \$5 per day or \$8 for 2 days, \$12 for 3 days

Applefest is produced by Northeast Expositions Inc. and is sanctioned by Apple Computer Inc. and The Boston Computer Society.

*Apple and Applefest are registered trade and service marks of Apple Computer Inc.



By Jon R. Voskuil

***“Beneath the rule of men entirely great,
The pen is mightier than the sword.”***

(Edward Bulwer-Lytton)

***“Words, words, words — I’m so sick of words!
I hear words all day through,
First from him, now from you!
Is that all you blighters can do?”***

(Eliza, in *My Fair Lady*)

“And God said, ‘Let there be light.’ And there was light.”

(Genesis)

***“But the words — oh, the words —
They don’t come easy.”***

(Mac Davis)

It seems that we’ve always had mixed feelings about words. On the one hand, their power and influence are undeniable. In the Genesis account, it was the power of the divinely spoken word that brought the world into being. And humanly spoken words, too, are acknowledged as being more powerful even than battles in shaping the course of history. On the other hand, along with Eliza (and especially when sitting through an evening of TV commercials), we get saturated with the endless flow of words that inundates us. And because words have been so cheapened by abuse, there are times when we can’t find the right ones to express the really deep and important things at the center of our lives.

Much of everyday living is spent in communicating with other people by means of words. And more and more of those words are being processed electronically. All of us are accustomed to electronic communication: The telephone has been a fixture of life for a long time. But not everyone is accustomed to being able to manipulate words in the ways that computers can — which puts even the best politicians to shame. Throughout more years of school than I care to think about, I pounded typewriter keys week after week, as term papers and lab reports came due. Erasable bond typing paper was one of my major budget items. Throughout eight of the past nine years since then, I’ve continued to pound the typewriter keys, as I’ve continued to produce ten to 20 pages of written material each week. (Certainly not much compared to what a professional writer would produce, but enough to make me weary of erasers and correction fluid.)

During this past year, however, I’ve been (a) writing more than ever before, and (b) enjoying it more. Item (b) is due largely to the computers which have been helping to remove much of the tedium from the writing process. Those of you who have used one of the many excellent text editor programs

available know what I’m talking about. Although I still use a typewriter occasionally (and even put pen to paper from time to time), it’s difficult to imagine going back to handwritten and typed manuscripts for serious writing. Writing with a quill and scroll by candlelight is romantic, but not my cup of tea.

As you’ve undoubtedly noticed, a major theme of this issue of *SoftSide* is computerized word processing. That’s just a little off the primary path of entertainment that we usually tread, but we think most of our readers (with the exception of one or two in our own programming department) see life as more than one big arcade game, and are looking for varied applications for their computers. Besides, many of us do find writing to be entertaining; and using a computer to do a lot of the hard work makes it more so.

Ironically, we’ve chosen this issue to announce that our recent experiment in computerized submissions procedures hasn’t turned out to be all we had hoped. (See “Outgoing Mail.”) Sometimes computers can get in the way more than they can help. At that point, one has to decide whether it’s worth it to put up with the computer’s clumsiness for the sake of some future goal. This is a legitimate approach; we do it with young children all the time. The alternative is to decide that the computers (or we) just aren’t ready for handing over that particular task yet. This is the decision we’ve reached on this one application of word processing in our precise situation. That’s not an indictment of computerized word processing, but an indication that (a) we really don’t have the right computer for the job, and (b) computers aren’t smart enough to do all the jobs that real people (especially editors) do.

Let me leave you with just one additional warning about the limitations of computerized word processing: Keep writing your love letters with quill and scroll by candlelight. ☺



From our readers

Due to the response we've received from our survey published in February, we wanted to share some of the comments with you in this month's Input.

"I like the articles as much or more than the programs. Especially Lance Micklus' My Side of the Page series and the articles by Scott Adams. The ones on the problems of marketing software and naming them are especially interesting! Hardware Corner on the problems of printer interfacing was good as were the disk articles.

After spending hours keying in *Quest* I had to go out and buy more RAM to make it work on my ATARI®. Aargh!!! The warmed over TRS-80® programs that are translated for ATARI® and don't take advantage of the available graphics are frustrating. Actually I have nothing but respect for the TRS-80® programmers who can squeeze so much out of their limited B&W graphics. BRAVO for *Microtext*."

Richard S. Waller
Seven Hills, OH

Editor's Reply: We try to publish the best software available for each system and we carefully select the best programs from the submissions we get. Since we receive few ATARI® submissions, we try to work around the problem. To get the ATARI® version of *Quest* to fit in 24K see "Bugs" in the December issue.

"On the whole, I would say that *SoftSide* is an excellent magazine. There are a few things I would like to mention though:

Arcade games, *TRS-Man*, *ABM Command*, although well written, are just too slow for BASIC. Simulations, e.g., *Titan*, *Dairy Farming*, *Bridge Contractors*, etc., seem to be the same thing only different settings.

Anything written by Edward Umlor is great! Keep up the good work!

One Liners are amazing at times. The problem: inconvenient to store & run separately — possible solution: Alter published ones so that instead of running infinitely as most do, have the current line transfer control to the next One Liner. These could then all be saved in one file.

Give quicker response to program submissions.

More Assembly programs & articles, *PROG-80* are you there??

More application programs & ideas. Printing a picture of the main screen in a program is a good idea — more of this.

How about some sort of chart which would show the best possible translation of a statement from one BASIC to another. Thanks for a great magazine."

Stephen Milliken
Randolph, MA

"Fully enjoy *SoftSide* magazines. I have dropped three other publications to start (TRS-80®) *SoftSide* DV. Articles covering given equipment are far more useful than ones in general that can not easily be converted to your machine. Keep it coming."

Roland E. Long
Cincinnati, OH

"I find typing programs a most useful help in learning to program — this is why I'm not buying the Disk Version, although it's tempting. All articles & columns for real beginners are better in your magazines than most — easier to understand."

"I have every issue of *SoftSide* from the very beginning. I am pleased to congratulate you on the present state of *SoftSide*. Although I've had my "trying times" with your subscription and shipping departments, I still look forward to receiving my magazines and ATARI® DV every month! Keep up the good work!!!"

Mansel H. Crowell
Vincentown, NJ

Editor's Reply: Thanks to all of those subscribers who have had subscription problems in the past for having patience. Bear with us as we make an attempt to go from our in-house subscription facilities to our use of a national subscription fulfillment house.

"Please consider reprinting *SoftSide Developing Database*. I missed the first issue which is no longer available! (This is the reason I'm not using it.)

I think your graph showing the advantages of the CV is great, but while those that don't have the CV are correcting (typos) I'm still waiting for the cassette to arrive! . . .

Spdyer, my man, Machine Head has got to be the No. 1 micro cartoon! Keep it coming! From someone who operates from the right side of the brain, Spdyer, I'm asking you to keep all those logical thinkers loose enough to allow some imagination — to enter the programming of the future!! Where can I get a T-shirt with the *SoftSide* logo on it? You people at *SoftSide* have more than customers out here, we're your FANS!!"

T.J. Pryts
Sharpsville, PA

"I have liked the changes you have made in the past year particularly: DV, *Microtext*, and *Database*. Also, the games have improved and mechanics of loading and operating have im-

proved. During the time of transition to DV, I often had to wait up to 3 weeks between receiving the magazines & disk but that seems to have worked out now too!"

M.C. Shannon
Ft. Worth, TX

Editor's Reply: As you DV and CV subscribers know, we are now mailing the magazine and media so they arrive at approximately the same time. Three cheers!

"If these comments are truly read, then know that *SoftSide* can stand some improvement. 1st) Magazines should arrive before they get to the store. 2nd) A toll-free phone number for customer service should be available. I've spent over \$100.00 trying to straighten out my subscription. 3rd) As of this date I am still trying to change my subscription from cassette version to DV.

Now for the good part: Your series "The Sensuous Programmer" is excellent. The *Database* series should be offered as a programming course in its completed form when it's finished. Thank you, I will continue my subscription for 1 more year, if I can get ATARI® DV!!!"

Steve Salkin
Palm Springs, CA

Editor's Reply: 1. The mailing of magazines to dealers and subscribers is done at the same time. Because the dealer packages go UPS and the other magazines go Second Class, there is a time difference in arrival. Nothing can be done to change this. 2. We do have a toll-free number for subscription inquiries. See the Table of Contents page. 3. It takes 6-8 weeks for the changes in subscriptions to take effect.

"Now have a subscription to ATARI® DV. You should include DOS 2 on each disk and make the cover program auto-booting!

When I receive each disk, there are usually some free sectors left. Why not fill the unoccupied space with something? You could fill it with back issue information, ads, or a catalog of items with prices of things offered by The Software Exchange.

Congratulations, February disk and magazine got to me by Feb 2; this is the fastest I've received them! I would like to be able to dial up CompuServe and download *SoftSide* programs; I'm set up for this now."

Editor's Reply: The reason the DOS has been taken off the ATARI® DV is due to a request from Atari. We cannot get permission to include the DOS. You must first boot up an ATARI® disk to use ATARI® DV.

As for the free sectors on your disk, we are going to be adding the K-Byters to the media and may possibly use your suggestion for the remaining space.

"For me the most useful feature of *SoftSide* has been the *Database* manager. Further developments in this area would be of value to me. I purchased DV subscription to save my time helping my son enter games from the magazine. However with additional utilities or editors and extensions of *Database & Microtext* I may consider continuing the DV for its value to me. The games — *Quest*, *Titan*, esp. — have been generally excellent. Suggest more Hi-Res use of Apple graphics. Some of your columns have been very good (Plaudits to Sensuous Programmer especially.)"

Eugene R. DeSombre
Chicago, IL

Editor's Reply: Because *SoftSide* relies on submissions, we are sometimes limited in what we publish. All you programmers out there — take note.

"If you are not going to publish a Model III DV, then could you at least make sure that the extra DV programs work on Models I/III? It is very annoying to get a disk with programs like *NEWBASIC* or *HELP* that I cannot use."

Jeff Metzner
Loudonville, NY

Editor's Reply: By the May issue, we hope to make available separate Model I and Model III versions of DV. We are using the survey to see how many of our subscribers have each system.

"1. You could possibly include a section about new users groups. 2. Couldn't you provide some extras for C/V subscribers — like giving all C/V subscribers a bonus cassette like that which you offer to new subscribers. Why not include "hybrid" and machine language programs on C/V *SoftSide*. You would be surprised what can be done with cassette! (e.g., *Cload* magazine & *Load 80* from Cload Inc. and *80 Microcomputing*, respectively.) Keep up the good work!"

Ben Davis
Amarillo, TX

Editor's Reply: We at *SoftSide* are in the process of compiling all the cards sent in for users groups. Watch future editions of *SoftSide*.

"I appreciate your efforts to support the ATARI® & I applaud your decision to finally put articles for the specific machines into their own "side" a la *Compute*. It makes your magazine much more useable. I certainly don't object to having programs for the other machines included in the same magazine, but realistically, there's not much useful dialogue that can occur until we understand each other's languages enough to make translations. How about a meaningful series of "Rosetta Stone" articles for that purpose? Obviously, a machine specific magazine would be nice, but I'm afraid the ATARI® edition would only be four or five pages long. (I think that because the ATARI® has been (erroneously) billed as merely a "non-serious", entertainment machine, that fewer of us are competent programmers. I know that I, for one, am still learning.) . . ."

Finally, several months ago, I wrote you a letter which was somewhat critical of your support of the ATARI®. I now wish to repent & applaud your efforts on our behalf. In recent months you

have presented some excellent programs for the ATARI® which are beginning to use ATARI® capabilities. . . ."

Bob Dewitt
Provo, UT

Editor's Reply: We are hoping to publish an article on how to translate from one system to another — a great suggestion made by a few people. These types of suggestions are what we hoped the survey would generate. Thanks.

"Your magazine is great — I wish it were published twice a month. The programs are accurate and I find the errors are usually my inaccurate typing. I would like more business applications — Alan Zett has been outstanding for me when solving problems. Since we're new owners, debugging has been somewhat difficult. Without him we would still be on our first program. Keep up the good work and thanks for your sincere cooperation."

Ronald S. Cohen
Lyndhurst, OH

"It's hard to make constructive comments when you are doing such a good job already, however, it might be interesting to see some articles about machine language/assembly language programming a la the Sensuous Programmer. Overall I think the mix of programs and articles is very good. I particularly like the monthly Hints & Enhancements. I'd like to see it expanded. Keep up the good work!"

Chilcott
Greensboro, NC

Editor's Reply: We look forward to good hints and enhancements being submitted along with developing some using our programming staff.

"My first computer was a TRS-80®. I hated the keyboard and graphics. I sold it and bought an ATARI® 800. My next purchase will be a printer. Actually the interface first. I dislike typing long programs however your disk version is too costly by a long shot. My disk is only one month old. I could use some articles on using it. I find no one is covering this area for newcomers. I would like to see complete addresses on letters printed so it would be possible to communicate with some of those other people out there. I have no quips about advertising. You need it, I even read it. Another suggestion is small program contests! Nothing big, look at *Softalk*. And maybe specifying a problem that can be solved on a computer as a routine with some/one solution given the next issue, and a new problem, etc.

Expand to include some assembly programming. Try to become an interactive magazine to be read, pondered, challenged, and acted on. Not just a program to be copied. Good Luck."

Larry Johnson
Bellingham, WA

"I like the idea of separate sections for ATARI®, APPLE, TRS-80®, and whatever other systems you decide to support. I hope you include other systems (such as the Color Computer) but please don't go back to separate magazines, just make one BIG one. I like seeing the info on non-ATARI® systems, and there are already magazines dedicated exclusively to each system. I think *SoftSide* has found a unique position in the computer magazine industry: pro-

moting cooperation, communication, understanding and "civilized behavior" among owners of "different" computers. Keep up the good work and good programs, and keep *SoftSide* together as one magazine. I sure wouldn't complain if it was thicker than BYTE!"

Mark Reid
S. Charleston, WV

"*SoftSide* is the only dependable source for good game software at a reasonable price. I have just subscribed to the cassette version as the programs longer than 16K take too much time to type in and debug. I have had very good results with most of the listings in *SoftSide*. Most of my software library is from *SoftSide*. Keep up the good work."

Lee S. McPherron
Thornton, CO

"I would prefer to see a magazine which has documentation and advertising with a separate code book for each computer."

Sharon Sloan
Bedford, TX

Editor's Reply: We have actually considered the use of a code book instead of printing pages and pages of line listings. The use of a code book would allow us more room for articles and reviews — maybe something for the future with *SoftSide*.

"This is the first issue I have seen of your mag. It was only luck that I saw your name in the ATARI® *APX* mag, otherwise I would have never known you existed. You have a very good mag (which I am going to subscribe to). I like the way your programs have descriptions above every few lines as to what the lines do. I would like to see a longer Q & A column with a postage PREPAID CARD in each mag so readers can easily send in questions. GREAT MAG!"

Ron Bieber
Beverly Hills, CA

Editor's Reply: Great idea about the post card. Will consider it in the future.

"Great magazine, gets better each month. More programs that use the special graphic (ATARI®) will help — its easy to copy TRS-80® & Apple.

Game Reviews are great — BUT NEED one or two pictures of the game in action!!!

Hope with the addition of TRS-Color that there won't be less of the three already: (Programs) Home Finance package with Graphics for Graphs would also be nice. What are you going to do when Microsoft comes out for ATARI®? Two Languages?"

John R. Low
Portland, OR

Editor's Reply: When we add the TRS-80® Color Computer we will not change our support of the Apple, ATARI® and TRS-80® I/III.

OUTGOING MAIL



by Nancy Lapointe

Greetings! As another issue nears completion and is about ready to go to press, the excitement at *SoftSide* peaks. We have been working very hard for some months to compile all the information found in this word processing issue. You will find comparative reviews on word processors for each system along with *Microtext 1.2* and two additional programs for each system. Enjoy!

Survey

We are starting to get back quite a few of your surveys from the February issue. Due to the response, we felt it would be worthwhile to use this month's Input column to share some of the comments with you. The answers to the survey will be very influential in deciding what will be in future issues of *SoftSide*. We need reader input because this magazine is published for you, the reader. So by printing some of the comments we have received, we hope to inspire those of you who haven't returned your survey to do so as soon as possible. Keep those surveys coming!

Media Subscribers

You media subscribers will be happy to know that starting with the March issue for the Apple and this issue for the TRS-80® and ATARI®, the K-Byters will be included on your respective media editions.

After many months of trying to get the magazine and media to arrive together, we finally succeeded. On the survey, the most frequent comment from our media subscribers was that they received their disks a few days before the magazines. It wasn't easy! But after finally finding the best sources of duplication and arranging our schedules, we have managed to

eliminate the time-lapse problem. Thank you all for bearing with the problem and having patience while we solved it.

SoftSide Authors

In the December issue of *SoftSide*, we asked that all program submissions be sent with documentation on disk or tape, using the *Microtext* word processing program. We have been living with this experiment for awhile now, and have come to realize that there are worse things in life than files of paper to be maintained. One of those things may be files of disks and tapes which can't be accessed without a spare computer (of the right breed) at hand.

THUS WE HAVE DECIDED FOR NOW TO RETREAT BACK INTO THE POPYRUS AGE AND ASK THAT AUTHORS ONCE AGAIN SEND ALL DOCUMENTATION ON PAPER. We encourage you to include text files on tape or disk as well, but this is not required.

A new Author's Guide will be completed shortly, detailing submission standards and procedures for those who may not be familiar with them. It's free for the asking — but a self-addressed, stamped, business-size envelope would be much appreciated.

Subscription Changes

For all of you who are upgrading your subscription to a media version of *SoftSide*, starting a new subscription, making an address change, or ordering back issues, please realize that you must have patience for approximately 4-6 weeks. It takes that long for the changes to be keyed in and to show up on the new mailing labels generated by our fulfillment house.

It probably is difficult for a subscriber of any magazine to imagine what is involved to produce and get you that magazine each month. In order for an issue to arrive at the beginning of a month, the printing and mailing must happen in the middle of the previous month. Therefore, any changes to subscription information keyed in after the first two weeks of the month will not show up on the labels attached to the magazine until the following month. If, after allowing this amount of time, you still have a question or doubt about your subscription please feel free to contact our fulfillment house in Broomall, PA at the toll-free number found on the Table of Contents page. You can also use this toll-free number to order subscriptions, make changes or order back issues.

On another note, I'd like to take this time to say how proud we are of our programmer, Rich Bouchard. A member of the Computer Science Team at Milford (NH) High School, Rich ranked second in the American Computer Science competition. His ranking is a result of two of the five contests in which the school will compete this school year. We are proud to have him on our team of programmers at *SoftSide*. Keep up the good work, Rich.

Well, enough of the news, info and gossip of *SoftSide*. I'll let you get on to the best part of *SoftSide* — the programs, articles, and especially this month's reviews. We'll be watching the mail for more surveys and your input.

Until next month, Happy Hacking.

Nancy Lapointe

Nancy Lapointe
Managing Editor



A Dozen Delights

By "J"

Back in 1947 (the year of my birth) a man named Alfred P. Morgan wrote the Preface to the second edition of his book, *The Boy Electrician*. In that Preface he wrote as follows:

Once upon a time, and this is a true tale, a boy had a whole railroad system for a toy. The train ran automatically, propelled by tiny electric motors, the signals went up and down, the station was reached, a bell rang, the train moved on again and was off on its journey around many feet of track to come back over the old route.

The boy viewed his gift with raptured eyes, and then his face changed and he cried out in the bitterness of his disappointment: "But what do I do?" The toy was so elaborate that the boy was left entirely out of the play. Of course he did not like it. His cry tells a long story.

The prime instinct of almost any boy at play is to make and to create. He will make things of such materials as he has at hand, and use the whole force of dream and fancy to create something out of nothing.

As I was growing up, I waged a single-minded campaign to keep *The Boy Electrician* from gathering dust on the shelf of our town's library. I succeeded pretty well. Finally my father, taking pity on my frayed library card, bought me my own copy of that wonder-filled book.

Its chapter titles induce waves of nostalgia as I look at them on the desk in front of me. "Voltaic Cells and Batteries" guided me through the construction of many a homemade battery (some more successful than others). "An Experimental Wireless Telephone" gave me the idea for a prize-winning Science Fair project in sixth grade on magnetic induction. "Electrical Measuring Instruments" taught me about galvanometers and Wheatstone bridges, concepts I later used (in the eleventh grade) to design sensitive measuring instruments necessary for a chemistry lab project.



My friend Charlie and I built our own radio following the instructions in "How to Build a Radio Receiver with a Crystal Detector." I even got a couple of working motors out of the chapter, "Electric Motors," and some first-hand experience with silver plating from the jam-packed twenty-first chapter, "Miscellaneous Electrical Apparatus." These many years later, as I page through that 407-page book, each illustration is as familiar as if I had drawn it myself last week.

I now have a son who is six years old. He's still a little young for *The Boy Electrician*. You might think that a book written 35 years ago about experimenting with electricity would be a little out of date for him anyway. Maybe it is. After all, this is the computer age. Who cares about making a microphone using a carbon rod and binding posts rescued from a discarded No. 6 Ignition Cell? (Who even knows what a No. 6 Ignition Cell is, anymore?) Who would ever want to make a Leyden jar, or a reflectoscope, or a night light with a little hook for hanging your pocket watch so that you can read the time in the dark?

After pulling that book off the shelf

this morning, and reading the 1947 Preface, I turned to the back of the title page and found the original copyright date: 1913. The book that had been so formative in my childhood had been written two years before my father was born. Undoubtedly, Alfred P. Morgan has died sometime in the past 35 years. As far as I know, he never did follow up on a possibility he expressed in the last sentence of his book: "And, as the future years bring new inventions and discoveries, no one now knows but that, some day, perhaps I shall write another "Boy Electrician."

The pace of change is unquestionably far greater now than when *The Boy Electrician* was written, or when I was a boy. The years that elapsed between those two periods saw nothing in the advance of scientific and technological knowledge, compared to that which has occurred since then. Even though the first computers were becoming a reality by the time Alfred P. Morgan wrote his second Preface, he probably didn't dream that so much could happen so quickly.

Now that such sophisticated electronic devices as Apples and ATARI® s

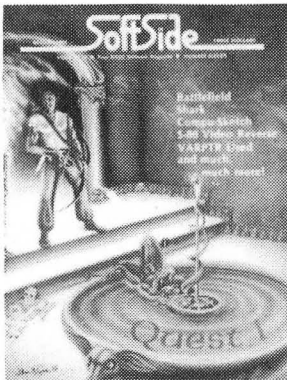
★ SOFTSIDE'S BEST FROM THE PAST ★

*"Don't it always seem to go,
That you don't know what you've got 'til it's gone..."*

Joni Mitchell

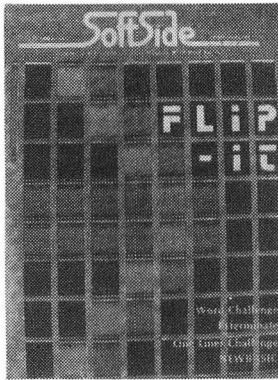
Sad to say, **Quest 1** and **Flip-It**, two of the most popular programs that ever appeared in *SoftSide*, are no longer available in print. But don't despair: *SoftSide* Publications is reprinting these two exciting games in a convenient, easy-to-use steno notebook format.

QUEST 1



Quest 1 is an exciting journey into an underground maze in search of treasure and adventure. A host of Undead and other unseemlies await the intrepid explorer. Armed only with sword and bow, you must rely on quick thinking to survive. And don't let the wraiths sneak up behind you!

FLIP-IT



Flip-It is an excellent implementation of the board game Othello™. Match wits with a formidable opponent: your computer.

ORDER YOURS NOW!

Our special introductory price for this outstanding software value is only \$2.95 for each booklet. Be sure to order now, before we come to our senses!

SoftSide

6 South Street Milford NH 03055

See page 16 for ordering information.

and TRS-80®s are becoming the everyday backdrop of our lives, has the simple boyhood romance gone out of experimentation and discovery? Is the computer another toy so elaborate that all we can do is sit back and ask, "But what do I do?" Has everything been done for us inside those chips and upon those printed circuit boards, so that there's little left that demands creativity and imagination? Has a person who has purchased a "canned" home computer utterly sold his soul to the devil of passivity?

Anyone who has experienced the agonies and the ecstasies of computer programming knows that the romance of experimentation and discovery is not dead. Certainly its form has changed. Involvement with a computer is more mental and less physical than involvement with an electromagnet made from a nail, a battery, and a few yards of wire from the hardware store. It is, I think, less intimate in a way — more detached, more ethereal, less nitty-gritty. Computers share many of the strengths and weaknesses of television, a powerful medium which at once exposes us mentally to more of reality than we would ever experience on our own, and at the same time lulls us into a physically passive rut. Just as television ought to be only a small part of a well-rounded life, among a great many other activities, computer programming should never be the sole creative outlet in a person's life. I'm going to see to it that my son winds electromagnets as well as typing keys and twisting game paddles.

Yet computers are stimulating in a unique way — even romantic, as hard as that may be for many people to believe. As I argued in my first column, the "inner reason" for existence which all computer programs share is to solve a problem of some kind, and that is surely a creative and sometimes romantic activity. I guess that if I weren't romantically involved with computer programming, I wouldn't have had any interest in writing this series of columns. And I presume that if you weren't involved in a similar way, you wouldn't be reading them. (The only people I can think of to whom that definitely wouldn't apply are friends and relatives upon whom I foist each issue of the magazine, saying, "Look! I'm the Sensuous Programmer!")

If, in any way, these articles have helped in the process of discovery and experimentation, then their purpose has been fulfilled. And who now knows but that someday, perhaps I shall write another "Sensuous Programmer."



My Side of the Page

By Lance Micklus

First this month, let's catch up on some of the news about our old friend, *The Mean Craps Machine*. Some time ago, I told you that Scott Adams and I had decided not to try to sell the game for the Model I/III and instead just concentrate on the Color Computer market.

Well, it turns out that Adventure International can now make a tape which will load on either the Model I, Model III, or the Color Computer. This means that after Scott makes the master tape, he can sell all three versions with no more effort or expense than a version for one system. So, *The Mean Craps Machine* will be available for Model I/III users.

I also made the comment some time ago that the secret to making money in this business is to milk a product for all you can get. If sales drop off and the product goes off the market, publish the program and squeeze a few more dollars out of it. One of the other programs I've brought back for the Color Computer is *Dog Star Adventure*. It's an excellent adventure-type program for beginners because it's challenging but not too much so. As a by-product of Scott's new tape recording technique, the original Model I version will now be available for both the Model I and the Model III.

I consider this to be a major development since it will open up the game market to some authors in both the Color and Model I/III market all at once. This increases the potential for sales and is a bigger incentive for more work to be done in writing computer games.

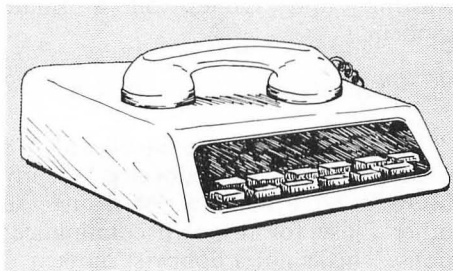
Just as interesting is that Big Five Software's disk games come on self-booting disks. What's more interesting is that their disk can boot on a Model I or Model III; single or double density; with 35, 40, or 80 tracks. It's nice to know somebody found a way around the operating system problems.

What happened to *Star Trek*? Unfortunately, Paramount Pictures and I were unable to agree on a license for

some of the new versions of the game I wanted to release. Among the problems is that after the *Star Trek II* movie comes out this June, it might be followed by a *Star Trek III* movie. Thus, my computer program couldn't be called *Star Trek III* until the third, as yet uncommitted, movie comes out.

Since a name change had to be made anyway, I decided to get out of the *Star Trek* loop and change the name to *StarFlite* which I am registering as a trademark. This should prevent problems with names in the future.

In case you missed some parts of the Getting a Bit Serious series, I may put them on my own BBS system if there is enough demand. The number of my BBS is 802-862-7023.



Modems, Modems, Who's Got The Best Modems

On my hit parade of the ten most frequently-asked questions is this one: Who makes the best modem? That's about as easy to answer as trying to figure out who makes the best computer.

I haven't seen all of the popular modems, but I've worked with many of them. And of the modems I haven't seen, I've heard enough comments from people who own them to form some opinions.

The function of a modem is to convert an electronic signal to an audio signal and then back to an electronic signal. An audio signal can be

transmitted over ordinary dial telephone equipment just like a human voice can. Thus, the modem interfaces the terminal and the telephone so that each can receive information in a form it can understand.

About a year ago, we stopped using the old acoustical coupler modems in-house and went to direct-connect modems. This change actually had no effect on users of our communication products, although it sure cleaned up my work space a lot. Until we went to direct-connect modems, we had a separate desk telephone for each phone line. This meant I had three telephones sitting next to me so I could use my modems on any phone line. After getting rid of all of the acoustical couplers, I had a ten-button key set installed and added another badly-needed phone line.

The acoustical coupler modems are the ones with the big rubber cups on the top. To use them, you dial the phone, and when you hear the tone, you place the receiver on top of the cups on the modem. One cup listens to the phone and the other cup is a speaker which talks into it, so there is no direct wiring into the phone. The advantage of this type of modem is that they are cheaper to manufacture and avoid any problems involved with connecting the customer's equipment to telephone company equipment.

On the negative side, the price advantage of acoustical coupler modems is now very small. The cost of direct-connect modems has come down dramatically in the past two years. Also, new changes in FCC regulations have put the phone companies in the position where they can no longer tell you that you can't connect your equipment to their system — provided your equipment is FCC approved.

The almost universal use of modular phone jacks makes it easy to install a direct-connect modem. However, with an acoustical coupler modem, you must have a standard headset. The cups will not fit such things as a

Trimline phone.

The biggest drawback to acoustical coupler modems is that they rely on the telephone headset itself. Telephone headsets (the things you hold up to your ear and talk into) use carbon microphones which are not known for their high fidelity. Some headsets are so bad that you can't even understand people when they talk in them, much less use them to send computer data.

You might be able to find a used acoustical coupler modem at an excellent price. (Make sure you have an RS-232 interface first.) If you do, buy it; but also understand what you're getting and make sure your telephone headset is in good condition.

The direct-connect modems come with various features depending on the kind you get. Basically, a direct-connect modem plugs directly into the telephone line rather than trying to talk into a headset. Its signal is not degraded by the carbon microphone in the headset so the fidelity of the transmission is better, which results in fewer data errors. Direct-connect modems are not bothered by other noises in the room and are the only practical way to set up unattended systems like FORUM-80, MOUSE-NET, CBBS, etc.

Other than the price, there are very few other drawbacks to direct-connect modems. Perhaps the biggest is that you must have a modular jack installed to connect it, but that shouldn't be a problem. In my case, since I'm using one of those business ten-button key sets for a phone, I had the phone company mount a separate set of modular jacks for each line coming into the office.

Just about all direct-connect modems offer auto-answer as a standard feature. Any of the hobbyist ones also offer auto-dial. Unfortunately, there are quite a few ways to auto-dial a modem and different manufacturers use different methods. If auto-dial is important to you, then you may find your selection of terminal software very limited.

There is a new group of modems starting to appear which call themselves "Smart Modems." According to Madison Avenue, it's a law that "Smart Modems" are better than "Dumb Modems." If you don't understand what you are buying, you may end up spending more money for a modem that doesn't do anything more or better than the cheaper dumb modems.

Smart modems use computer logic chips in them so that they can actually read the data you give them before it is

transmitted. Certain types of characters are interpreted as instructions to the modem and are not sent. Thus, the smart modem is controlled by transmitting certain characters to it rather than by turning on and off the various handshake signals which most terminals have. If you don't have the handshake signals to control the modem, this is the only way to do it. But many terminals and computers do have the handshake signals and you'd only be wasting your money to throw away your handshake signals and pay extra to control the modem "the hard way."

I have yet to hear of a smart modem made by anyone which can be used in a "practical" way for unattended operation. What you find is that you have to use the handshake signal to control your communications in the unattended environment and don't let anybody tell you otherwise. So the smart modems either include or will install at extra charge the handshake signals. This, of course, defeats the whole purpose of the smart modem which is to control the thing without handshake signals.

If you have a TRS-80® Color Computer, which has no handshake at all, then a smart modem might be worth the extra money. But if you own a Model III and want to use your modem to run a bulletin board, you'd only be wasting your money on a smart modem.

The Standard of Excellence

I have two Anderson-Jacobson 1258 modems. One is connected to my MOUSE-NET bulletin board and the other I use for my own communications. This is not a hobbyist modem. I think the AJ-1258 cost \$995, although they can be leased for around \$55 per month.

The AJ-1258 is a direct-connect modem with auto-answer. It does not auto-dial. Its main feature is that it will run both standard 300 baud and 1200 baud in either Bell 202 or Vadic modes. It automatically selects the standard by some magic process I've never completely understood.

Another feature I like is the complete set of indicator lights on the modem. You know exactly what's going on. In fact, when somebody calls my MOUSE-NET board, I can tell if they're using one of my ST80 programs and often know which ST80 they're using just by looking at the indicator lights on the modem.

More important than the indicator lights is that the AJ-1258 correctly uses

the handshake signals according to the RS-232 standards. Some of the hobbyist modems don't and this makes for a lot of software compatibility problems.

In terms of performance, the AJ-1258 has been answering the phone on the bulletin board for almost a year now. It's taken calls from all over the country on good and bad phone lines with all kinds of modems on the other end. It really does get you through.

The major drawback to the AJ-1258 is its high price. Also, the first time you try to connect it, you'll have a lot of fun fiddling with the more than 40 switches and jumpers you can change around.

Radio Shack Modems

The new Modem II by Radio Shack won't be out in any quantity until you read this. Since I don't have one for evaluation now I can't say anything pro or con.

The Modem I seems to be a good seller for Radio Shack — almost to the point of phasing out the Level II acoustical coupler modem. This shouldn't be surprising since the Modem I is \$50 cheaper than the Level II modem and it's direct-connect too. As everybody knows, direct-connect modems are better. Even I said so earlier. I should have made one exception: The direct-connect Modem I is worse than any I've ever seen except my old Pennywhistle 103. I would call it a tie.

The Modem I seems to work fine on a local phone call to a local computer if the telephone line quality is good. Naturally, when Radio Shack demonstrates their Modem I, they call their local Compuserve number and can usually get a good connection. Then comes the trouble. After the customers buy the modem and call their own timesharing system, they may find that they get a lot of data errors. If it is a long distance call, they may find that they can't even establish a carrier at all.

Almost all of the complaints I get from ST80 users regarding data errors are from Modem I owners. I have also found that Modem I users on my MOUSE-NET bulletin board usually have a fair amount of trouble. I suspect some of them can't even get a carrier going, but I have no way of knowing this.

To check this out, my wife Dianne and I decided to have a little fun at Radio Shack's expense. We were down in northern New Jersey for Christmas so we decided to go to some of the

computer stores in Paramus and act like your basic dumb customer.

First we stopped at GEM Electronic whose window signs boasted that it had microcomputers and low prices. I guess their idea of a microcomputer is different than my idea. They did not know where the Radio Shack Computer Center was but suggested that we continue down Route 17 and we'd probably run into it — which we did.

In the Paramus Radio Shack Computer Center, Dianne and I asked like dumb customers to see the communications stuff they had for a Model III. I asked if I might be able to try it with a terminal program I had with me in the car called *ST80-III*. (I had a heck of a time keeping a straight face.)

It became quite clear that our salesman was new and he went straight by the book — no non-Radio Shack vendor software allowed. We were shown *Videotext* using the Modem I on CompuServe and it seemed to work just fine. I also asked if I could try their *Videotext/Model III/Modem I* on my CompuServe account but it took the permission of the store manager before I could do it. Then I decided to push my luck again. After some careful prompting, I talked our salesman into letting me call my own bulletin board here in Vermont on a credit card call. I made three tries at it and I couldn't even get a carrier.

Of course I was given the story about how the phones are screwed up and that the computer I was calling was not run by Radio Shack. I finally leveled with the salesman and told him who I was. It didn't matter. He never heard of me. I should have known that when he said he never heard of an *ST80*.

True, this is only one incident with one salesman who had only worked at the computer store for three months and didn't know too much about communications. But what happened to me is exactly what other people say happened to them at other stores with other salespeople. Also, as others had told me, if it's not a local call, the Modem I may not work at all. It didn't work on my system whose quality I know is excellent.

Just to finish the story, Dianne and I finally met the manager of the Paramus Radio Shack Computer Center who knew who I was based on my *Star Trek* game. Just before we left, we asked if he knew of any other computer stores in the area, but he said he couldn't help us because he didn't know of any. So, Dianne and I left and as we got back to the car, Dianne said, "Look!" Two hundred feet down the road was a sign that said COMPUTER

UNIVERSE. The Radio Shack store manager had to drive by COMPUTER UNIVERSE every time he left the parking lot. There is absolutely no way you can drive north in the south bound lane of Route 17 even if you are trying to commit suicide.

COMPUTER UNIVERSE proved less exciting than GEM Electronics. The magazine rack did not have *Kilobaud*, *80-Microcomputing*, *SoftSide*, *80-US*, *Creative Computing*, or *Byte*. There was no software from Instant Software, Sensational Software, Ramworks, Adventure International, Acorn, Big Five, or SBSG. We left after a few minutes for Burger King. True, they don't sell computers, but at that point we needed some "junk food."

Lynx

About a year ago, Emtrol replaced their original Lynx modem with a new model that would work on either the Model I or the Model III. This version of the Lynx includes its own built-in RS-232 board, auto-answer and auto-dial. On the Model I, it plugs directly into the back of the keyboard or into the screen printer port of the expansion interface. An extra printed circuit board is included for the Model III. The board plugs into the I/O bus on the bottom of the Model III. Switches on the board make the I/O port addresses selectable. This avoids interference with the Radio Shack built-in RS-232. It comes with the *ETERM* program which is excellent for tape users but probably won't do the job for disk users.

Although I did not write *ETERM*, the Lynx is based on some of the suggestions I made to Emtrol when the new Lynx was being designed. Unfortunately, my Model III *ST80-III* was ready before the new Lynx so I had to second-guess them. My guesses were

correct, but without the actual modem in hand, there was no way to make sure *ST80-III* and the Lynx would work perfectly.

To make a long story short, I blew it. In order for the Lynx to work on the Model III, you need to turn the I/O bus on. This is a simple software instruction that can be done from BASIC. Unfortunately, *ST80-III* does not turn on the I/O bus correctly. This will be fixed in the next upgrade. In the meantime, you must turn on the I/O bus from BASIC before running *ST80-III*. There is no problem with this on the Model I.

The advantages of the Lynx depend on the machine to which you connect it. On the Model I, the Lynx RS-232 is far more reliable than the Radio Shack one. Lynx is software compatible with most terminal and host software on the Model I. On the Model III, you may have to change the port addresses of your terminal program.

For *ST80-III* users, there's a public domain program called *LYNX/BAS* which does this for you. If *LYNX/BAS* was not on your *ST80-III* disk, call my bulletin board at 802-862-7023 and download it.

Very little auto-dial support is available for the Lynx right now. However, auto-dial will be available for the Lynx in the next upgrade of *ST80-III*. Other than that, auto-dial is only available in the *ETERM* program.

On the negative side, the Lynx seems to have a reputation for being a little deaf. I have no way to check this out but I suspect that it is true. In the few cases I've heard of where the Lynx seemed to need a little more poop, a simple adjustment usually did the job. In any event, the Lynx is vastly superior to the Radio Shack Modem I and I doubt you'll have much trouble with it. Plus Emtrol customer support is excellent.



When does your *SoftSide* subscription expire?



49007STANTG97*B00P1281OCT 82
1102691 017
GEOFFREY STANTON
97 BAYSIDE COURT
KALAMAZOO, MICH. 49007

The last five characters (three letters for month, two numbers for year) on the top line of your mailing label will tell you when your subscription ends.

For more information, write:
SoftSide
515 Abbot Drive, Broomall PA 19008

Microconnection

The Microconnection I'm referring to here is the one with auto-answer and auto-dial. It requires an RS-232 board to use and is a direct-connect modem. Thus, it will work on the Model I, Model II and the Model III.

If there is one good thing I can say for the Microconnection, it's this: When the going gets rough, if a Microconnection can't make it, then nothing will. In the area of sensitivity and noise elimination, it will perform as good or better than my \$995 Anderson-Jacobson. Indeed, it would be the perfect modem were it not for one problem — software compatibility.

It seems that the handshake signals that the Microconnection wants are non-standard just for normal operation. So you will have to patch your terminal program or make up a special RS-232 cable to get it to work unless you buy a smart terminal program also marketed by the Microconnection people. This is unfortunate because the Microconnection is getting a bad reputation for being software incompatible. But assuming you don't mind making software patches, the Microconnection is a good buy and does deliver the performance.

BIZCOMP

I have the BIZCOMP 1084 Direct Connection Smart Modem with the 06 option installed. The 06 option gives you all the handshake signals but at additional cost. It is auto-answer and auto-dial and requires an RS-232 in the computer to operate.

Unlike the other modems discussed so far, the BIZCOMP will auto-dial with any software. Since it does not need any handshake at all, you can even use it on the TRS-80® Color Computer. It's small, reliable, and easy to use. It also has a lot of bad features.

Noticeably lacking are any indicator lights. Not even a power-on lamp is provided. As a result, I once spent ten minutes trying to get it to work only to find out that I forgot to plug in the power supply.

To use it for unattended operation, you must add the 06 option, then jumper some of the RS-232 wires around. This defeats the whole selling point of any smart modem, which is that it can be controlled without handshake signals.

D.C. Hayes Stack Smart Modem

This is one modem I haven't seen, but I've heard a lot of comments about it. Unlike the BIZCOMP, the D.C. Hayes Modem does have indicator lights on it and includes handshake signal processing at no extra charge. It is far more intelligent than the BIZCOMP 1084. For example: It will do touch-tone dialing, set the number of rings required before answering, and even turn off the response codes.

On the negative side, I have not been impressed with the customer support from D.C. Hayes. I've had several people set up their MOUSE-NET system with the D.C. Hayes and all of them had trouble. None of them were able to get any information from Hayes regarding the handshake signals used in their modem. The solution actually came from Don Johnson who runs a MOUSE-NET down in Florida. He figured out how to rewire the pins correctly so they would work with MOUSE-NET.

I have mixed feelings about the Hayes. Personally, I think Hayes is trying to treat the TRS-80® as if it were an Apple. They definitely are relying on their Apple reputation in the TRS-80® market. On the other hand, I am biased against smart modems and haven't seen the Hayes. If I had one to use for a while, I might change my mind. It is priced competitively with dumb modems, yet gives you the smart

modem capability when you need it. I guess my advice on this one is to check it out very carefully and to decide for yourself.

Other Modems


When I first started running my own bulletin board system, I used the old faithful US Robotic. It is now sitting on top of my AJ-1258 as a backup modem. In terms of price, it's not the value it used to be. But it is an old faithful friend that's really good for a BBS system.

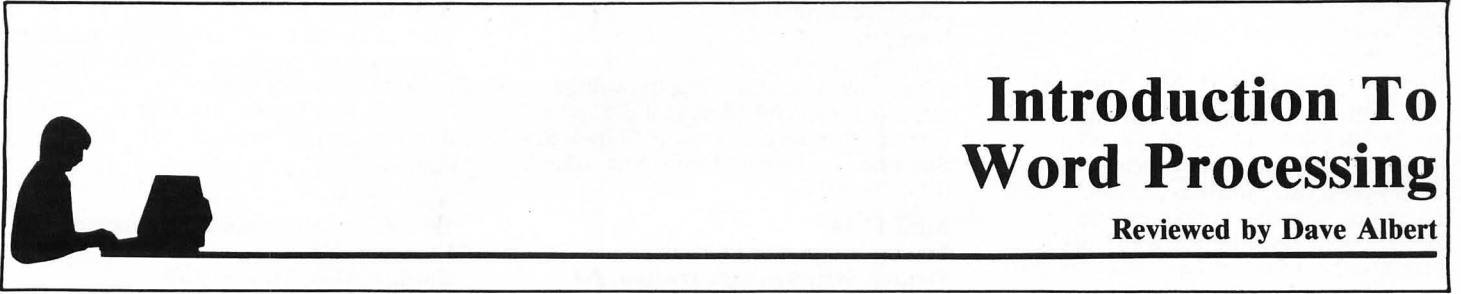
There are also a couple of other modems available that include their own terminal program in a built-in ROM. I'd think twice about these. Someday you may want to use some other terminal program and then you can be stuck because you'll also have to replace the ROM modem with a standard one. In the long run, you might be better off getting a no-ROM modem and your favorite terminal software to drive it.

Choice

There are certainly a lot of other modems around that I haven't mentioned, but the ones I have covered are the most popular makes. The best pick really depends on what type of a system you have, and what kind of features you want. If you are going to use your modem for a bulletin board system, stay away from the smart modems. If you are going to use your computer as a terminal, then compare carefully. You may find that the terminal software you like will give your favorite dumb modem all the power you want rather than going with a smart modem and a dumber terminal program. In general, it is better for the computer to directly control the modem with handshake signals than to try to run it by remote control.

My own personal choice is the Anderson-Jacobson 1258. I want the auto-answer with either 300 or 1200 baud capability. It is compatible with all my software. It has lots and lots of lights and is very reliable. It does not auto-dial but that's not a problem to me; my phones have speed dialing on them anyway. All I do is dial two digits and the telephone company does the rest.

Most people have different requirements than I do, so I would imagine that the AJ-1258 would not be your best choice. But whatever you decide on for a modem, I hope the information I've given you will help you make the right choice. 



Introduction To Word Processing

Reviewed by Dave Albert

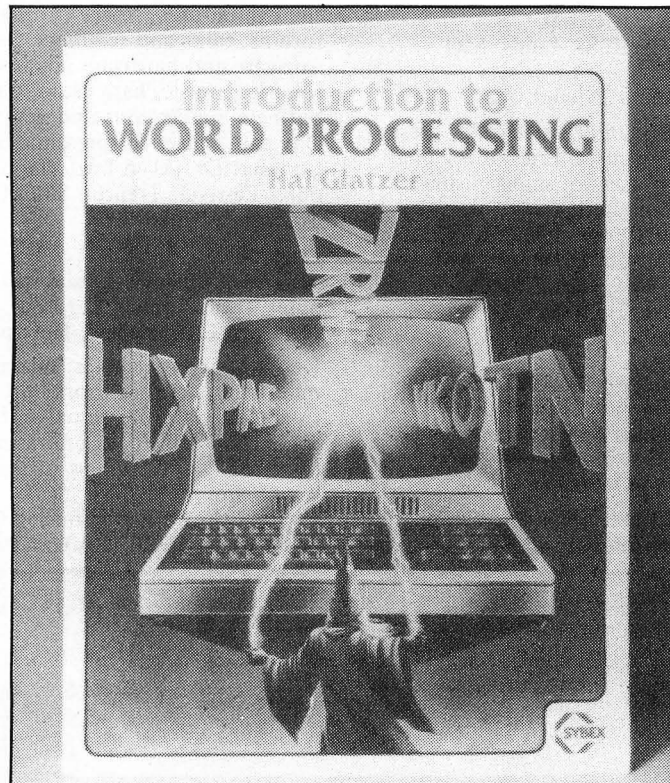
by Hal Glatzer (Sybex) Suggested retail price: \$12.95

First things first. This is NOT a book about microcomputer word processing applications. To my knowledge, such a book does not exist. However, if you are unfamiliar with word processing applications in general, then this book may be of considerable interest to you. It is a well-written, concise discussion of word processing, from stand-alone electronic typewriters to mainframe computers.

Author Hal Glatzer has done his homework. Furthermore, he has managed to avoid the many possible pitfalls in writing about the burgeoning field of electronic text processing. He does not get lost in technical details that would bore the novice, nor does he succumb to the temptation to riddle his manuscript with technical jargon comprehensible only to the slide rule set. Instead, he has written a book that anyone can understand. His language is simple and very straightforward. His expressed (and achieved) aim is to help non-technical people to get a handle on the possibilities and capabilities of equipment in the new field called word processing.

Introduction to Word Processing opens with a discussion of what "word processing" is, and how it can be valuable to a wide variety of people. Glatzer obeys the cardinal rule of showing, not telling, in this section. He seems to suggest that word processing capabilities can be valuable to almost anyone, a point this reviewer would be reluctant to contradict. He then undertakes to explain what options are available in the realm of letter-crunching.

The options Glatzer discusses are: electronic typewriters, stand-alone word processors, microcomputers, and mainframe and mini computers. At no point does Glatzer suggest one type of system is better than the others; he simply details the advantages and disadvantages inherent in each. The




points he covers are ease of use, cost, cost effectiveness, versatility, and a host of others. Again, he only mentions a few brands, and he does not ever really express an opinion about those he mentions.

With the groundwork laid, Glatzer then tackles actual operation of the various word processing systems. He does not specifically tell one how to use a word processor, but he does go into considerable detail about general operation of the different types, including discussion of the most commonly used functions of word processors. Furthermore, he discusses the various storage media available for the equipment, formatting for printing, and different types of printing available with word processing equipment.

Finally, Glatzer gives some good practical advice about getting hands-on experience with word processing before

running out and dropping a bundle on some machinery. His points are well taken, and he backs up his suggestions with an adequate appendix listing various sources of information about word processing. He also provides a substantial glossary at the end of the book, which will help any potential buyer wade through otherwise hard-to-comprehend industry literature.

In sum, this is a fine introductory guide to word processing. It will not help one to decide what system is best, although it probably will be quite useful in narrowing down one's choices. Glatzer is to be lauded for not discussing specific equipment, nor trying to act as word processor guru in dispensing his wisdom. His writing is quite easy to understand, and his approach is figuratively to take the reader by the hand and lead him merrily into a hitherto confusing world. It is, in all, a job well done. 

CALENDAR

April 2-4

The Second Annual Eighty/Apple Computer Show

New York Statler Hotel, New York, NY

Show will feature products and services for the TRS-80® and Apple computer systems. More than 100 exhibitors will display hardware, software, books, magazines, supplies, and accessories.

Contact: Ken Gordon, Kengore Corp., 3001 Rte. 27, Franklin Park, NJ 08823, (201)297-2526.

April 15-17

Computer Showcase Expos

A.J. Cervantes Hall, St. Louis, MO

April 23-25

Miami Expo Center, Miami, FL

Sponsored by The Interface Group, these two shows are part of a series of regional public shows for users of small business and personal computers.

Contact: Peter B. Young, The Interface Group, POB 927, 160 Speen St., Framingham, MA 01701, (800)225-4620. In Massachusetts, (617)879-4502.

April 15-18

The Second Southwest Computer Show and Office Equipment Exposition Market Hall, Dallas Market Center, Dallas, TX

April 22-25

New York Computer Show and Office Equipment Exposition

Nassau Coliseum, Uniondale, NY

These shows will feature mini- and microcomputers for business, education, government, industry, and home use. Data processing and word processing equipment, office machines, computer peripherals and office supplies will be displayed. General admission is \$5.

Contact: National Computer Shows, 824 Boylston St., Chestnut Hill, MA 02167, (617)739-2000.

April 17-18

Trenton Computer Festival

Trenton State College, Trenton, NJ

This seventh annual festival will feature an outdoor flea market, indoor commercial exhibit area, *Space Invaders* contest, forums, talks and seminars. Sessions on software and hardware applications, robotics, music, ham radio, education, business applications and new languages. Day care center for preschool children.

Contact: Dr. Allen Katz, Trenton Computer Festival, Trenton State College, Trenton NJ 08625.

April 20-22

American Business Equipment and Computer Trade Show

Hartford Civic Center, Hartford, CT

Designed for executives, managers, administrators and advisors in the business, financial and professional world, this trade show will focus on business operations today and preview office systems of the future.

Contact: Key Productions, 410 Asylum St., Hartford, CT 06103, (203)247-8363.

April 20-22

D-COM

Hynes Auditorium, Boston, MA

A trade show featuring products and services compatible with Digital Equipment Corporation's products, D-COM will involve vendors and users.

Contact: Ron Davies, D-COM Inc., 7312 Burdette Court, Bethesda, MD 20817, (301)469-7650.

April 22-23

1982 Rocky Mountain Data Processing Expo

Currihan Hall, Denver, CO

Will feature displays of various product lines and services for offices. Conferences will include sessions on training, databases, EDP auditing, programmer productivity, and management concerns.

Contact: Mark Cramer, Industrial Presentations West, 3090 S. Jamaica Court, Suite 304, Aurora, CO 80014.

If you or your organization are sponsoring or know of an event you think would be of interest to *SoftSide* readers, please send complete information to:

SoftSide Publications

Calendar Editor

6 South Street

Milford, NH 03055

Be sure to include complete information concerning dates, location, subject matter and a contact name, address, and phone number.

SOFTSIDE ORDERING INFORMATION

FORM OF PAYMENT

USA

VISA, MasterCard, certified checks, money orders and personal checks are accepted.

Canada/Mexico

The preferred method of payment is by VISA or MasterCard. A bank check is acceptable if it has been preprinted for payment in U.S. dollars. No personal or company checks accepted.

Other Foreign Orders

Payment must either be by a bank check drawn on a U.S. bank payable in U.S. dollars or by affiliated bank credit cards of VISA or MasterCard.

GUARANTEE

All software is guaranteed to load and run. If you experience difficulties with the product within 30 days, it may be returned for replacement. Send your properly protected tape or disk to the attention of the Customer Service Representative and include your name, address, and the reason it is being returned.

LIABILITY

All software is sold on an as-is basis. **SoftSide** assumes no liability for loss or damage caused or alleged to be caused directly or indirectly by products sold or exchanged by them or their distributors, including, but not limited to, any interruption in service, loss of business or anticipatory profits or consequential damages resulting from use or operation of such software.

PRICES

Prices are subject to change without notice. We are not responsible for typographical errors.

Unless otherwise noted in a published advertisement, the following prices are in effect as of this issue:

	USA Canada	Mexico APO/FPO	Other Foreign
SoftSide Magazine (yr)	\$30	\$40	\$62
CV (year)	\$75	\$95	\$125
(6 mo.)	\$39	n/a	n/a
DV (year)	\$125	\$145	\$175
(6 mo.)	\$64	n/a	n/a
Adventure of the Month Month (6 mo.)			
Cassette	\$29	\$35	\$41
Disk	\$49	\$55	\$61
Parsec	\$20	\$21.50	\$25
Envyrn	\$200	\$202	\$210
Diversions	\$60	\$70	\$85

BACK ISSUES

Minimum order for magazines only — 3 issues. There is no minimum order for magazine/media combinations. Price includes shipping to the 48 states only. Alaska, Hawaii, Puerto Rico, APO/FPO, and ALL foreign orders — postage is additional.

ALL Foreign orders and all magazine/media combination orders — Order directly from **SoftSide, 6 South St., Milford, NH 03055.**

SAVE / on Software and Hardware for TRS-80[®]

CHEXTEXT[®]

Let your TRS-80[®] do the proofreading on your SCRIPSI[™] text files!

Features of this program include:

- Complete dictionary maintenance including the addition and deletion of words.
- Menu driven for ease of operation.
- Spelling Checker
- FREE expanded dictionaries available, depending on your drive storage capabilities.

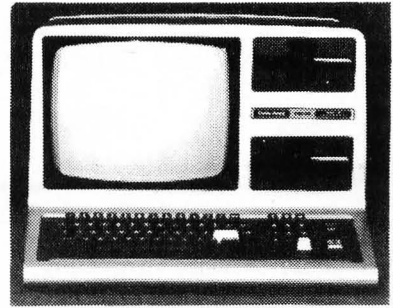
NEW LOWER PRICE \$59.95

MODEL III PRICE LIST

(All 48K with TANDON drives)

1 single sided 40 track drive	\$1695.00
2 single sided 40 track drives	1895.00
1 dual sided 40 track drive	1820.00
2 dual sided 40 track drives	2145.00
1 single sided 80 track drive	1845.00
2 single sided 80 track drives	2175.00
1 dual sided 80 track drive	1995.00
2 dual sided 80 track drives	2395.00
with RS-232 C Add	115.00

NOTE: These Model III computers contain Apparat installed disk drives and memory. They are warranted by Apparat, Inc. for 90 days



MISCELLANEOUS SUPPLIES

DISKETTES

Double density-soft sectored-replacement guaranteed. Spindle/Hub protected. (5 1/4" only)

Verbatim Datalife 5 1/4" 40 track	\$24.95
Apparat's No Name 5 1/4" 40 track	\$19.95
Verbatim Datalife 8" model II	\$39.95

PAPER

9 1/2"x11" blank white, tractor feed paper, full box 15# or 20#	\$24.95
14 1/2"x11" green bar, tractor feed paper, full box	\$34.95
3 1/2"x15/16" tractor feed mailing labels	\$19.95

OTHER

5 1/4" plastic library case	\$ 1.95
8" plastic library case	\$ 4.95
5 1/4" Flip-sort	\$18.95
8" Flip-sort	\$31.95
16K memory kits	\$19.95

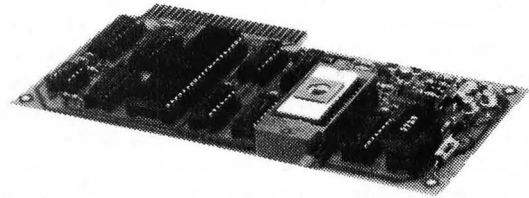
APPARAT'S PROM BLASTER

An eprom programmer for all 25 X X and 27 X X chips. TRS-80[®]

MOD I & III	\$149.00
CABLE	\$ 17.95

BUS EXTENDER

mini version with 2 card slots and no enclosure	\$ 69.95
---	----------



SPECIAL - FREE GRAFTRAX

with EPSON MX-80 Printers

MX-80 **\$499.00**

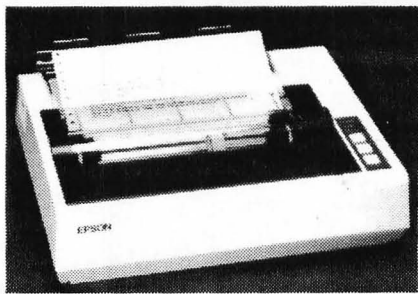
MX-80F/T **\$575.00**

MX-100 **\$775.00**

Printer Cables **\$24.00**

(Specify Computer Type)

**WE ALSO STOCK OKIDATA & NEC PRINTERS
CALL FOR PRICES**



ASSORTED ITEMS OF INTEREST

MICRO CLINIC, Mod I \$24.95, Mod III \$29.95

The ultimate in memory & disk diagnostics

MEAL MASTER, Mod I & III Disk \$24.95

meal planning & grocery shopping aid

FLEXTXT/80 (requires Graftrax) \$34.95

utilize the additional features of the MX-80

printers under model I & III scripsit

MICRO ACCOUNTING SYSTEM, \$479.00

Interactive G/L, A/R, A/P & checkbook manager

UNI-TERM/80, \$89.00

Universal terminal program that takes advantage

of the extended NEWDOS/80 commands

BASIC BETTER & FASTER BOOK, \$23.95

NEW LOWER PRICES ON TANDON DISK DRIVES

Complete with power supply, chassis & configured for TRS-80[®] Model I or III

- Single sided 40 track
 - Dual 40 or Single 80 track
 - Dual sided 80 track
 - **Special - 2 Dual sided 80's**
- (ADDS ALMOST 1.5 MEGABYTES OF STORAGE TO A D. D. MOD I OR A MOD III)

MODEL I DOUBLE DENSITY PACKAGE

Converts a standard TRS-80 Model I to Double Density.

Apparat Doubler & NEWDOS/80 V.2.0.

\$278.00 value for only **\$219.00**

Doubler alone **\$129.00**



NEWDOS 80 Version 2.0

The most sophisticated DOS ever produced for the TRS-80[®] Models I and III. It provides the user with "MAINFRAME" power on a "MICRO".

Some Features available are:

- Jobstream Control Language
- Mod I/Mod III Diskette interchangeability
- Double Density Support on Model I
- Pagenation of BASIC listings on the screen
- Basic program single stepping
- Dynamic variable manipulation
- Multiple array sorts with BASIC CMD
- Complete technical support provided

All this plus much more for only

\$149.00

InfoWorld

Software Report Card

NEWDOS/80 Version 2.0

	Poor	Fair	Good	Excellent
Performance	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Documentation	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
Ease of Use	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Error Handling	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>



Apparat, Inc.

"On-going Support for Microcomputers"

4401 S. Tamarac Pkwy. • Denver, CO 80237 • (303) 741-1778 • (800) 525-7674

Scrpsit & TRS 80 are a registered trademark of Tandy Corporation

Freight F O B Denver call for shipping charges Foreign Orders shipped Air Freight

Word Processing: An Art In Transition

By Tom Stanton



Another phase of word processing, the printed word, is beginning to pass as the dominant form of information storage and retrieval. Print has lasted over 500 years, during which time it spread knowledge in an unprecedented fashion, providing economical copies of classics and original works to whomever could afford them. At no other time in history has so much information been so quickly exchanged. Save, perhaps, for the present.

Yet electronic storage and retrieval is but a change in technologies. Printing will remain, just as handwriting remains, but it will become as uneconomical as handwriting in producing copies. Ultimately, that is all that our "information revolution" involves: copies. A larger number of available copies, whatever their origin, depends on our desire for them, and the cost per copy.

The Written Word

The first word processors were the scribes, that legion of men and women who followed their masters and wrote of God, business, taxation, and law. They organized their writing skills into a middle-level profession that lasted until the 16th century. With their mistakes, shorthand, and inventions the scribes shaped our alphabet.

Our earliest written records seem to be packing slips. Recent archeological discoveries of symbols stamped on clay vessels have led to speculation that those jars were packing slips, shipped with goods to account for quantity, item, and price. Other discoveries link these symbols with sealed vessels containing stones inside. Archeologists think that stone stamps were cut, pressed into wet clay jars, then sealed inside as a "hardcopy" record of a shipment. These clay vessels have been found almost exclusively in grain houses or aboard sunken ships, and some have remained unbroken, their contents visible only with X-rays.

Stone, clay, and paint were the favorite media for permanent records;

while bamboo, bark, or palm leaves provided semi-permanent media. Alternatives to clay tablets were sought almost from the beginning, and more effort put into developing thin plant or animal skins upon which to write. Egyptian papyrus, in particular, and Formosa "rice paper" were cut from the pith of trees, making them light, portable, not too easily damaged, and economical enough for its purposes.

During this same period, both the Egyptians and the Chinese developed new tools for writing, the former using reed pens, the later a brush. Both of



A sixteenth century portrait of Johann Gutenberg.

these led to more precise character form and the slow evolution of "standardized" writing, that is, to a recognized style as opposed to personal interpretations. Using this foundation the written word made progress.

During the reign of Trajan, in A.D. 105, when his forum was built and each scribe wrote on a hinged board covered with wax (called a codex), T'sai Lun announced to the Chinese emperor, Yuan Hsing, the development of paper. It would take 1000 years before the first paper mills started production in Europe, although printing pictures with wood blocks had already been in-

vented. However, the commercial connection between paper and print could not be completed in the Orient; the ideogrammatic forms favored calligraphy, and it was not until this century that the technology existed to create economical printing for these languages.

After the fall of Rome, Christian monks took over the task of the scribes. The monks retreated with the writing skills and perfected them; they created our upper case and lower case; they established the Roman alphabet as the dominant form in Europe; and they created the book, an enduring means of information storage and retrieval. In every country, using various "national" styles of writing, our alphabet and its end use, the book, grew to maturity. The monks worked in small rooms, three or four to a room, using quill pens and various ink recipes on parchment which was dried animal skin, scraped and polished to smoothness. These rooms opened on a large central room, called the scriptorium, where the work was supervised, proofread, and then passed on to others who gilded or painted decorative initials and pictures on the parchment. The work was done in stages, each monk forwarding the pages until a book was bound and finished.

Naturally these books were highly valued in their day. The fact that possibly ten copies of any work existed was "miraculous," at least by 11th century standards. The monks, who had originally created only liturgical works or approved classics, shared their skills with secular craftsmen, and by Gutenberg's time, the middle of the 15th century, there were large "commercial" scriptoriums as well as religious ones.

The commercial scribes joined guilds and bound themselves with other craftsmen in related fields to form a thriving and powerful group that had seemingly limitless demands for its products. The introduction of paper to 10th century Spain, while not im-

mediately significant to the future of scribes, at least assured them of reasonably-priced materials upon which to write.

The Japanese had already invented block printing in A.D. 760, during the time of Charlemagne in Europe. Kibino-mabi, a Chinese-trained scholar, returned to his native Japan after 20 years of study and brought with him, as some believe, the invention of kata-kana, the oldest and simplest form of Japanese writing. He also supervised the production of scrolls printed from blocks of wood. This first printing, for the Empress Shotoku, consisted of four charms, or dharani, on scrolls about as wide as the column you're reading and 22 inches long, all placed in small wood pagodas. The dharani had been issued after a terrible outbreak of smallpox in Japan, as celestial insurance against another such outbreak.

Block book printing in Europe, however, had been stifled by the powerful guilds, whose scribes and artists had laws passed banning the duplication of images. Printing's first flowering in Europe was from a brisk business in playing cards, followed by devotional images printed by "underground" artists of the day.

It would seem that this period was a most unauspicious time to introduce printing. Yet, it was the perfect time economically, and for one important reason: the Renaissance had created a demand for more books than scribes could ever hope to produce. Many inventors sought to perfect printing: Coster, from the Netherlands, produced a few crude prayer pamphlets using type cast from clay; and Waldvogel, a Czechoslovakian goldsmith in Paris who invented "artificial writing" in 1446, are two whose names and inventions have been documented. It remained for Gutenberg to create the effective printing technology to take advantage of the abundance of paper produced (at prices well below the cost of any parchment) by a number of mills that had come into existence since 1150.

The Printed Word

Gutenberg did not "invent" printing. He invented the processes that made it economical, practical, and marketable. To make type he borrowed from his craft as a goldsmith and produced steel rods upon which letterforms were filed. These rods were then stamped into copper or brass plates. The plates, with their letters in relief, were fitted to a mold. A mixture

of lead, tin and antimony was poured in, and when it had cooled and was opened, the type was removed, the metal dressed, and more letters were cast.

Although it sounds like an agonizingly slow process, it was a very quick and efficient by 15th century standards. Gutenberg also made other contributions including a good printing ink and the use of a screw press adapted from the winemakers.

At first printed books were declared semipermanent at best by the guild of scribes, but it took only 50 years to change that. By 1500, printing had already gone through its classic period in Germany and Italy, and emerged as a full-blown industry whose number of workers would exceed those of the scribes in another half a century.

The art of typography in printing took up where calligraphy had left off



Fashioned Lettering styles such as the above example are frequently used in modern magazines and newspapers. Few styles other than Old Fashioned instantly suggest antiquity.

in writing. The type you're reading now has its origins in the Italian calligraphy favored during the Renaissance. Called humanist after the scholars who used it, the letterform was adopted by Italian printers as an alternative to Gutenberg's dark Gothic type, derived from the German "national style." The Italians, and in particular a displaced Frenchman named Nicholas Jenson, nurtured and perfected our present roman alphabet. Some have called Jenson's roman type the most perfect roman ever made.

Though typography flourished, the printing industry itself changed very little. Up until the 1840s books were printed by hand in large rooms containing as many as 15 presses, each producing two sheets a minute. It was the invention of continuous rolls of paper that spurred printers to apply power to their presses.

In 1814, the *Times* of London began using a power press, and from this development evolved the modern daily newspaper. During this same period, a new method of printing, lithography, was developed, but the full scope of this discovery would not be felt until after the Second World War.

By the 1880s power presses and continuous paper production were commonplace, yet type still had to be set by hand, letter after letter, much as it was in Gutenberg's day. Many devices for setting type had been tried, including one that cost Samuel Clemens (Mark Twain) considerable money before it folded.

The first real advance came in 1886 with the introduction of the Linotype machine in America and the Monotype in England. The Linotype, as its name suggests, forms a line of type from reusable type molds, casts it, and then returns the type molds to their proper place in the machine. A Monotype, on the other hand, casts each letter individually. Concurrent with these developments was the invention of the typewriter.

As these new technologies took the place of the more labor-intensive ones, print thrived as it never had in previous centuries. Our use of printed paper was measured by metric tons, and the many magazines and newspapers published between 1890 and 1945 document this expansion.

During the Second World War, however, a new kind of technology was being developed. The computer, with an ancestry nearly as long as printing, but more obscure, was being conceived and nurtured with War funding. After the war, the favorable climate for computers grew with the economy of the United States and Europe. The "business machine" and computer-phobia became a part of American folklore.

While the computer was being developed, lithographic printing began to replace the 500-year-old relief printing, and the photo-litho process helped create a new means of setting type. The phototypesetters developed during the 1950s were retro-fitted linotypes, whose type molds carried a letterform on negative film which was exposed through lenses onto photo paper, developed, and used to make photo-litho plates.

In the 1960s this process was refined, and a rotating drum with the characters on film spun about a strobe light, which fired on command whenever the correct letter appeared, exposing the letter onto photo paper. The Photon Corporation produced one of the first machines for the U.S.

Government Printing Office.

At the same time, IBM was perfecting a new type of business machine they called a "word processor." The first models developed for the type-setting industry used a typewriter instead of a photocopier for its output. A typist recorded a job during keyboarding on magnetic tape, which could be stored and used over again without re-keyboarding. IBM later introduced a similar machine to business offices, but such devices had not yet become an economic necessity.

With the commercial introduction of microchip technology, a new breed of typesetter was created. These first machines used the microchips to store programs that automatically justified (made columns even), hyphenated words, and determined where a line would end without operator assistance. They used magnetic tape or hard disk storage at first, but as floppy disk technology became practical and the cost of the chips went down, all storage and computing was done by the machine itself. They were the forerunners of word processing machines for the business office.

Some word processing systems had been in place since the mid-70s, but almost exclusively in larger corporations. With the development of the

microcomputer and related software, these technologies have filtered down far enough so that most anyone in business can afford some kind of word/data processing.

The Electronic Word

Printing will not be replaced by the electronic word before the end of this century. It may never be wholly outmoded, for there are new printing technologies, among them the ink jet, that may bring it back to a kind of prominence. But it will never dominate as it has for the last five centuries.


The microcomputer, armed with a word processing program and a printer, is capable of making digitalized copies complete with updates, at a cost that will one day be competitive with print. For short runs of, for example, 100 copies of a 20 page report, it is quite competitive. Still, this is only an overlapping of forms and technologies that compete to do the same thing. The power of the electronic word exceeds print because it does not need hard-copy replication. The data can be stored without once printing it, information can be sent nearly anywhere, and the cost of the raw materials involved is considerably less than those used for printing.

Remember, too, that I'm speaking

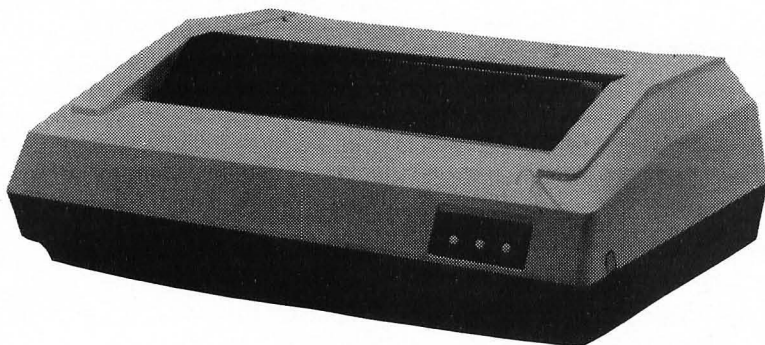
solely about the microcomputer. The deliberate exclusion of mainframes reinforces the fact that the electronic word not only can, but will dominate as our chief source of information. Books will not disappear; they will become rarities devoted to topics worthy of such presentation; instead of the present shabby products that consume millions of acres of vegetation, we may once again have books made "the way they used to be made."

The future of the electronic word will depend on the people who use it. We may develop a global library, translatable into any language, or we may simply develop a collection of our trivia. The question is one of quality.

With the ease of reproduction, will our future copies of data be looked upon as we look upon old books or the monumental writings in stone left us by the ancients? This revolution in information has been taking place since we started writing. We now have a new tool, as crude as we conceive an ancient pen might have been, but no more powerful than that pen for one reason: the hand that guides it.

Perhaps the greatest question we must answer is whether we are ready for this new tool, or even worthy of it. You are the pioneers. You are shaping those answers. 

The Little Printer with Big Features



Specifications

Comet 80-Column Dot Matrix Printer

Print Features	Number of columns	80 col. max.
	Print Speed	125 CPS
	Print Direction	Single-directional and Bidirectional Switch Selectable
	Throughput Speed	63 LPM at Bidirectional Printing
	Form Feed Speed	10 LPS
	Character Format	7 x 7 Dot Matrix
	Character spacing (max. number of columns per line)	Normal Width 10 CPI (80) Double Width 5 CPI (40) Compressed Font 16.5 CPI (132) Compressed Font Double Width 8.25 CPI (66)
	Line Spacing	6 LPI
	Print Width	203 mm (8") max.
Forms	Width	113 mm to 254 mm (4.5" to 10.0")
	Total Thickness	0.05 to 0.28 mm (0.002" to 0.011")
	Number of Copies	Original + 2 copies nominal
Form Feed	Method	Sprocket Feed
	Form Loading	Either bottom or rear

The **Comet-I** is the lowest priced printer that we sell. The **Comet-I** has standard features that are optional on many other higher priced printers. **Comet-I's** print speed is an impressive 125 cps bidirectional, and print features include 4 character sizes, 4 different alphabets for international use, a high-resolution 7 x 7 dot-matrix, and a print compression capability that enables the **Comet-I** user to switch between 5, 8.25, 10 and 16.5 CPI and to print up to 132 characters per 8-inch line. This paper saving feature is very desirable in today's data processing market.

You'll find that the quality, performance and cost economy of this printer puts it ahead of any other competitive model in its price category, and it's ready for immediate delivery.

Comet-I Printer, with Centronics-compatible interface
#09-259005H \$289.00

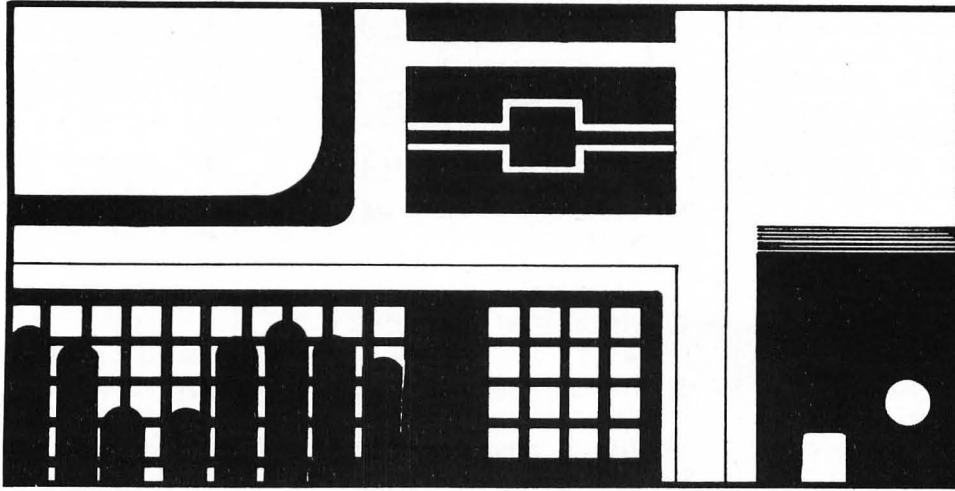
Comet-I Printer, with RS-232C interface
#09-259006H \$289.00



The Software Exchange

14 South St., Milford, NH 03055 (603)673-5144
TOLL FREE OUT-OF-STATE 1-800-258-1790

MICROTEXT 1.2



by Jon Voskuil

“Microtext 1.2 is a word processor program for a 16K Apple (with Applesoft), ATARI®, or TRS-80® (Model I or III).

In the December, 1981, and January, 1982, issues of *SoftSide*, we published the first two parts of a simple word processor program. Here we add the third (and probably last) main module of the program, which gives it line-editing capability.

A few minor modifications have also been made to other parts of the program. The complete line listings are given; lines which have been added or changed are indicated, so that you need only add these if you have previously typed in the other installments. Please observe that the “Program Documentation Textwriter” subtitle has been dropped, and it is NOT necessary to use *Microtext* in submitting documentation for programs; see *Outgoing Mail* for more information.

Upon running, *Microtext 1.2* displays a mostly blank screen with an instruction summary line at the top or bottom. You can either start typing, or load a previously saved file from disk or tape. (Please see the note below concerning the Apple version’s tape operation.) Capitalization of characters on the Apple and TRS-80® Model I versions is accomplished by preceding individual characters to be capitalized by an “@” symbol, and words to be capitalized by a double “@@”. Normal use of the Model I’s shift key inserts the “@” in front of individual characters; the “@” key itself must be used to insert the double symbol necessary for word capitalization. On the Apple, the shift key is not designed

to operate in this way; pressing ESC once or twice before the letter or word to be capitalized will do the trick.

The Apple and ATARI® versions use the CTRL key to access editor and system functions, while the TRS-80® version uses the CLEAR key. Holding down CTRL, or pressing and releasing CLEAR, and then pressing S, L, R, P, or E, will access the save, load, review, printout, or edit functions. Although not mentioned in the command summary on the screen, pressing CTRL-Q or CLEAR-Q will quit the program.

Saving and loading files is simply a matter of answering the questions about the medium to be used (tape or disk) and, if disk, the file name. Once you have entered a file name, it will be used as the default until you specify another one or exit the program: Just press RETURN/ENTER when asked for the file name. This simplifies repeated saves during entry of a long document. The Apple disk read/write routine has been modified to make it much faster than before, with the sacrifice of not being able to use standard quotation marks within the body of the text. (An apostrophe, or single-quote, is automatically substituted if the double-quote key is pressed.) If you insist upon retaining the use of normal quotation marks, you’ll have to put up with the slower disk I/O. This tradeoff is not necessary with the ATARI® and TRS-80® versions.

The review function causes the computer to return to the beginning of the text in memory and scroll through it to the end. During this scrolling, you can

press any key to pause. Then, pressing the spacebar will cause one more line to be displayed; pressing RETURN/ENTER will cause the scrolling to continue; and pressing E will enter the editing mode.

In the editing mode, you can move a cursor up and down through your text, to locate any line which you want to edit or delete. This movement is accomplished with the up- and down-arrow keys on the ATARI® and the TRS-80®, and with the I and M keys on the Apple. You have four options while in the editing mode: Pressing ESC (ATARI® or Apple) or CLEAR (TRS-80®) will exit to the review mode; pressing D will delete the line at the cursor; pressing X will delete everything from the cursor to the end of the text; and pressing ENTER/RETURN will allow you to edit the line at the cursor.

If you choose to edit a line, the screen will first clear, and then display a number of lines of text with a gap of several lines in the middle. The cursor will be positioned at the beginning of the line you have chosen to edit, and you can proceed to type in a new line to replace the old one. The new one can be shorter than the original, or may occupy multiple screen lines. Any part of the original line that you want retained must be retyped: Whatever you type in will replace the entire line. When you have finished entering the new text, press CTRL-F or CLEAR-F (not ENTER/RETURN, unless you want a carriage return in the text itself). The computer will check to see if the text lines need to be rearranged, and then return you to the review mode.

The printout function allows you to send your text to a printer, after selecting margins, line spacing, and (in the case of the Apple and the Model I) the case conversion option. Pressing RETURN/ENTER in response to the offered options will select the default value indicated. The case conversion option allows the text to be printed from the Apple or Model I just as it appears on the screen, in all upper case and with the embedded @ symbols; or else to convert it (a bit sluggishly, using a BASIC routine) to normal mixed case for a final printout.

Program Notes

The DIMension statement in line 120, and the CLEAR in line 110 of the TRS-80® version, reserve memory for the strings which hold the text. The numbers listed work for a 48K system (40K ATARI®) with DOS booted, but will need to be adjusted downward if your system has less available memory. All REM lines may be deleted without effecting the program's operation, to gain more memory space. Many of the individual program lines could also be squeezed together on multiple-statement lines; this kind of packing has been avoided for the sake of clarity, but would increase available memory and possibly execution speed.

If you have a TRS-80® Model III, some of the listed lines should be changed as follows:

Line 50: Omit "":GOTO 100"

Lines 59-74: Delete

Line 7050: Delete

Line 7130: Change to "LPRINT TAB(LM);P\$;"

Line 7140: Delete

Line 7590: Change to "LPRINT TAB(LM);PP\$;"

Line 7600: Delete

If you have an Apple, and have tried using the tape I/O routines included in previous versions of *Microtext*, you have undoubtedly noticed that they simply do not work. The STORE and RECALL statements are not designed for use with string arrays, even though they SEEM to work if you don't disturb the computer's memory between program runs. (That's what fooled us into thinking that they did work.) We are working on a method of reading and writing string arrays using tape, which doesn't involve the time- and memory-consuming process of converting the string arrays into integer arrays first. We hope to come up with a solution for publication in the May issue.

APPLE™

```

*****
$  APPLESOFT BASIC  $
$  'MICROTEXT 1.2'  $
$  AUTHOR: JON R. VOSKUIL  $
$  (C) 1982  SOFTSIDE  $
*****

```

Title page. (Lines 20 and 40 are changed.)

```

10 TEXT : HOME
20 VTAB 8: PRINT TAB( 8)"M I C
   R O T E X T  1 . 2"
30 VTAB 12: PRINT TAB( 12)"BY J
   ON R. VOSKUIL"
40 VTAB 14: PRINT " COPYRIGHT 19
   82 SOFTSIDE PUBLICATIONS"
50 FOR Z = 1 TO 3000: NEXT Z: GOTO
   100

```

Mixed-case printout conversion routine.

```

59 REM LOWERCASE PRINT RTN
60 X$ = "": IF PP$ = "" THEN 74
62 LOCK = NO:LC = N1: FOR K = N1 TO
   LEN (PP$):CC = ASC ( MID$
   (PP$,K,N1)): IF CC = N32 THEN
   LOCK = NO
64 IF CC < > N64 THEN 70
66 IF LC = NO THEN LOCK = N1
68 LC = NO: GOTO 72
70 PRINT CHR$ (CC + N32 $ (LC AND
   CC > N64 AND CC < N91) $ ( NOT
   LOCK));:LC = N1
72 NEXT
74 RETURN

```

Initialization. (Lines 130 and 210 are changed.)

```

99 REM INITIALIZATION
100 HOME : ONERR GOTO 20000
110 POKE 35,22
120 DIM L$(1000)
130 QUOT = 34:ESC = 27:BKSP = 8:R
   TN = 13:SPC = 32:B$ = CHR$
   (8):CR$ = CHR$ (93)
140 CHAR = 1:NO = 0:N1 = 1:N32 =
   32:N64 = 64:N91 = 91
150 LN = 1
160 D$ = CHR$ (4)
170 PRINT D$;"MON C,I,0": HOME
180 LWID = 38
190 V = 1:H = 1
200 VTAB 23: FOR I = 1 TO 39: PRINT
   "-";: NEXT : PRINT

```

```

210 VTAB 24: HTAB 1: INVERSE : PRINT
   " ^SAVE ^LOAD ^REVW ^E
   DIT ^PRINT";: NORMAL
220 VTAB V: HTAB H

```

Input loop. (Line 505 is changed. The original line 890 has been deleted here, but may be retained if preferred.)

```

499 REM INPUT LOOP
500 GET C$:C = ASC (C$)
505 IF C = QUOT THEN C = 39:C$ =
   "" : GOTO 740
510 IF C = ESC THEN C$ = "@":C =
   64
520 IF C = RTN THEN C$ = CR$
639 REM BACKSPACE
640 IF C < > BKSP THEN 720
650 IF CHAR < 2 THEN 500
660 PRINT B$;" ";B$;
670 CHAR = CHAR - 1
675 IF CHAR < 2 THEN L$(LN) = ""
   : GOTO 500
680 L$(LN) = LEFT$ (L$(LN), LEN
   (L$(LN)) - 1)
700 GOTO 500
719 REM CTRL CHARACTER
720 IF C < ESC AND C < > RTN THEN
   2000
739 REM END OF LINE
740 CHAR = CHAR + 1: IF CHAR < LW
   ID OR C = SPC OR C = RTN THEN
   760
750 BOSUB 1000
759 REM ADD TO STRING
760 L$(LN) = L$(LN) + C$
779 REM RETURN
780 IF C < > RTN THEN 880
820 PRINT C$
840 LN = LN + 1:L$(LN) = "":SLOC =
   0
850 CHAR = 1
860 GOTO 500
879 REM PRINT ON SCREEN
880 PRINT C$;
899 REM UPDATE SPC POINTER, CHK
   FOR LINE END
900 IF C = 32 THEN SLOC = CHAR: IF
   CHAR = LWID THEN LN = LN + 1
   :L$(LN) = "":CHAR = 1:SLOC =
   0: PRINT
920 GOTO 500

```

Subroutine to break line at a space and to initialize the next line.

```

999 REM JUSTIFY & INCR LINE
1000 IF SLOC = 0 THEN PRINT : GOTO
   1100

```



```

1020 HTAB SLOC
1040 PRINT SPC(LWID - SLOC); PRINT
1050 IF SLOC = LWID - 1 THEN 110
    0
1060 L$(LN + 1) = RIGHT$(L$(LN)
    ,LWID - 1 - SLOC)
1080 L$(LN) = LEFT$(L$(LN),SLOC
    - 1)
1100 LN = LN + 1
1120 PRINT L$(LN);
1140 CHAR = LEN(L$(LN)) + 2
1150 SLOC = 0
1160 RETURN

```

Subroutine to process command codes.
(Lines 2050 and 2600 are new.)

```

1999 REM PROCESS CTRL CHARS
2000 V = PEEK(37) + 1; H = PEEK
    (36) + 1
2050 IF C = 6 THEN RETURN : REM
    CTRL-F (BACK TO EDIT RTN)
2100 IF C = 18 THEN GOSUB 3000:
    REM CTRL-R
2200 IF C = 19 THEN GOSUB 4000:
    REM CTRL-S
2300 IF C = 12 THEN GOSUB 5000:
    REM CTRL-L
2400 IF C = 17 THEN TEXT : END
    : REM CTRL-Q
2500 IF C = 16 THEN GOSUB 7000:
    REM CTRL-P
2600 IF C = 5 AND LN > 1 THEN I =
    LN - 1; VV = V; VTAB VV - 1; GOSUB
    9000: GOSUB 3000: REM CTRL-E

2900 GOTO 210

```

Subroutine to review entered text. (Lines 3020, 3110, 3125, and 3190 are changed. Original lines 3130, 3140, and 3150 have been deleted.)

```

2999 REM REVIEW TEXT
3000 VTAB 24: HTAB 1: PRINT SPC(
    39)
3010 VTAB 24: HTAB 1: PRINT "PRE
    SS SPACE BAR TO PAUSE";
3020 HOME : SPEED= 240
3030 FOR Z = 1 TO 500: NEXT Z
3040 IF LN = 1 THEN 3210
3050 FOR I = 1 TO LN - 1
3060 PRINT L$(I)
3070 IF PEEK(-16384) < 127 AND
    NOT STP THEN 3200
3080 STP = 0: SPEED= 255: POKE -
    16368,0
3090 VV = PEEK(37) + 1
3100 VTAB 24: HTAB 1: POKE 35,24
    : POKE 34,23: PRINT

```

```

3110 PRINT : PRINT " RETURN:CONT
    SPACEBAR:STEP E:EDIT ";
3120 GET X$: X = ASC(X$)
3125 IF X = 69 THEN GOSUB 9000:
    GOTO 3000
3160 IF X = 13 THEN 3190
3170 IF X < > 32 THEN 3110
3180 STP = 1
3190 SPEED= 240: POKE 34,0: POKE
    35,22: VTAB VV: HTAB 1
3200 NEXT I
3210 PRINT L$(LN);
3220 V = PEEK(37) + 1; H = PEEK
    (36) + 1
3230 SPEED= 255: RETURN

```

Subroutine to save text to disk or tape.
(Lines 4075, 4085, 4088, and 4150 are changed. Line 4210 is retained for historical purposes only; the tape save routine doesn't work. See article for more information.)

```

3999 REM SAVE TO DISK/TAPE
4000 VTAB 24: HTAB 1: POKE 35,24
    : POKE 34,23: PRINT
4020 VTAB 24: HTAB 1: PRINT "SAV
    E TO TAPE OR DISK? (T/D/ESC)
    ";
4030 GET X$: PRINT
4040 IF X$ = CHR$(ESC) THEN 42
    20
4050 L$(0) = STR$(LN * 10000 +
    SLOC * 100 + CHAR)
4060 IF X$ = "T" THEN 4190
4070 IF X$ < > "D" THEN 4020
4075 F1$ = F$
4080 INPUT "FILE NAME: "; F$
4085 IF F$ = "" AND F1$ = "" THEN
    4080
4088 IF F$ = "" THEN F$ = F1$
4090 INPUT "INSERT DISK AND PRES
    S RETURN "; X$
4100 PRINT D$;"OPEN "; F$
4110 PRINT D$;"DELETE "; F$
4120 PRINT D$;"OPEN "; F$
4130 PRINT D$;"WRITE "; F$
4140 FOR I = 0 TO LN
4150 PRINT CHR$(34);L$(I); CHR$
    (34)
4160 NEXT I
4170 PRINT D$;"CLOSE "; F$
4180 GOTO 4220
4190 INPUT "START RECORDER AND P
    RESS RETURN "; X$
4200 FLASH : PRINT " SAVING ";: NORMAL
4210 STORE L$
4220 PRINT

```

```

4230 POKE 34,0: POKE 35,22
4240 RETURN

```

Subroutine to load text from disk or tape. (Lines 5065, 5075, 5078, and 5150 are changed. Original lines 5140 and 5160 have been deleted. Line 5230: see note on line 4210.)

```

4999 REM LOAD FROM DISK/TAPE
5000 VTAB 24: HTAB 1: POKE 35,24
    : POKE 34,23: PRINT
5020 VTAB 24: HTAB 1: PRINT "LOA
    D FROM TAPE OR DISK? (T/D/ES
    C) ";
5030 GET X$: PRINT
5040 IF X$ = CHR$(ESC) THEN 52
    50
5050 IF X$ = "T" THEN 5210
5060 IF X$ < > "D" THEN 5020
5065 F1$ = F$
5070 INPUT "FILE NAME: "; F$
5075 IF F$ = "" AND F1$ = "" THEN
    5070
5078 IF F$ = "" THEN F$ = F1$
5080 INPUT "INSERT DISK AND PRES
    S RETURN "; X$
5090 PRINT D$;"OPEN "; F$
5100 PRINT D$;"READ "; F$
5110 INPUT L$(0)
5120 GOSUB 5290
5130 FOR I = 1 TO LN
5150 INPUT L$(I)
5170 NEXT I
5180 PRINT : PRINT D$;"CLOSE "; F
    $
5190 FOR Z = 1 TO 500: NEXT Z
5200 GOTO 5250
5210 INPUT "START RECORDER AND P
    RESS RETURN "; X$
5220 FLASH : PRINT " LOADING ";:
    NORMAL
5230 RECALL L$
5240 GOSUB 5290
5250 PRINT
5260 POKE 34,0: POKE 35,22
5270 GOSUB 3000
5280 RETURN
5290 L$ = L$(0); L = LEN(L$)
5300 CHAR = VAL(RIGHT$(L$,2))
    : L$ = LEFT$(L$,L - 2)
5310 SLOC = VAL(RIGHT$(L$,2))
    : L$ = LEFT$(L$,L - 4)
5320 LN = VAL(L$)
5330 RETURN

```

Subroutine to print the text file in memory to a printer.

```

6999 REM PRINTOUT ROUTINE

```

```

7000 HOME : VTAB 6:LIN = 0
7010 INPUT "LEFT MARGIN? (DEFAULT
      T = 10) ";X$:LM = VAL (X$):
      IF LM < 1 THEN LM = 10
7020 PRINT : INPUT "RIGHT MARGIN
      ? (DEFAULT = 70) ";X$:RM = VAL
      (X$): IF RM < 1 THEN RM = 70
7030 PRINT : INPUT "LINE SPACING
      ? (DEFAULT = 2) ";X$:LS = VAL
      (X$): IF LS < 1 THEN LS = 2
7040 LL = RM - LM
7050 PRINT : PRINT "CONVERT TO L
      OWERCASE UNLESS PRECEDED
      BY @? (DEFAULT = NO) ";: GET
      X$:UC = 1: IF X$ = "Y" THEN
      UC = 0
7060 PR# 1
7070 HOME : PRINT :P$ = "":CR =
      0:I = 0
7080 I = I + 1:P$ = P$ + L$(I)
7090 IF RIGHT$(P$,1) = CR$ THEN
      CR = 1: GOTO 7110
7100 IF LEN (P$) < 255 - LWID AND
      I < LN THEN 7080
7110 GOSUB 7500:CR = 0
7120 IF I < LN THEN 7080
7130 PRINT TAB( LM);: IF UC THEN
      PRINT P$;: GOTO 7150
7140 PP$ = P$: GOSUB 60
7150 PRINT : PR# 0
7160 GOSUB 3000
7170 RETURN
7500 L = LL
7510 IF LEN (P$) > LL THEN 7550
7520 IF NOT CR THEN 7640
7530 LP = LEN (P$): IF LP < 2 THEN
      PP$ = "":P$ = "": GOTO 7590
7540 PP$ = LEFT$( P$,LP - 1):P$ =
      "": GOTO 7590
7550 C$ = MID$( P$,L,1): IF C$ =
      " " THEN 7580
7560 L = L - 1: IF L > 0 THEN 755
      0
7570 L = LL
7580 PP$ = LEFT$( P$,L):P$ = RIGHT$(
      P$, LEN (P$) - L)
7590 PRINT TAB( LM);: IF UC THEN
      PRINT PP$;: GOTO 7610
7600 GOSUB 60
7610 FOR J = 1 TO LS:LIN = LIN +
      1: PRINT " ": NEXT J
7615 IF LIN > 59 THEN FOR J = 1
      TO 66 - LIN: PRINT : NEXT J
      :LIN = 0
7620 IF LEN (P$) > LL THEN L =
      LL: GOTO 7550
7630 IF CR AND LEN (P$) > 0 THEN
      7530

```

7640 RETURN

Subroutine to readjust lines in memory so that they fit properly on the screen after editing. (This routine is wholly new.)

```

7999 REM TEXT REJUSTIFY RTN
8000 HOME : VTAB 5: PRINT "RE-JU
      STIFYING TEXT. . ."
8010 LIN = EL:LN = LN - 1
8020 P$ = "":CR = 0:I = EL - 1
8030 I = I + 1:P$ = P$ + L$(I)
8040 IF RIGHT$( P$,1) = CR$ THEN
      CR = 1: GOTO 8060
8050 IF LEN (P$) + LEN (L$(I +
      1)) < 256 AND I < LN THEN 80
      30
8060 GOSUB 8500
8070 IF NOT CR THEN 8100
8079 REM END OF PARAGRAPH
8080 X = I + 1 - LIN: IF X = 0 THEN
      8150
8089 REM MOVE LNS DOWN IN ARRAY
8090 FOR J = I + 1 TO LN:L$(J -
      X) = L$(J): NEXT J:LN = LN -
      X: GOTO 8150
8100 IF I < LN THEN 8030
8110 L$(LIN) = P$
8120 CHAR = LEN (P$) + 1:SLDC =
      LEN (P$)
8130 FOR I = LIN + 1 TO LN:L$(I)
      = "": NEXT I
8140 LN = LIN
8150 RETURN
8500 L = LWID
8510 IF LEN (P$) > LWID THEN 85
      50
8520 IF NOT CR THEN 8630
8530 LP = LEN (P$)
8540 PP$ = LEFT$( P$,LP):P$ = ""
      : GOTO 8590
8550 C$ = MID$( P$,L,1): IF C$ =
      " " THEN 8580
8560 L = L - 1: IF L > 0 THEN 855
      0
8570 L = LWID
8580 PP$ = LEFT$( P$,L):P$ = RIGHT$(
      P$, LEN (P$) - L)
8590 L$(LIN) = PP$
8600 LIN = LIN + 1
8610 IF LEN (P$) > LWID THEN L =
      LWID: GOTO 8550
8620 IF CR AND LEN (P$) > 0 THEN
      8530
8630 RETURN

```

Subroutine to edit lines of text. (This routine is wholly new.)

```

8999 REM EDIT LN SUBRTN
9000 IT = I: POKE 34,0: POKE 35,2
      2: IF I > 20 THEN V1 = 21: GOTO
      9040
9010 VTAB VV: HTAB 1:V1 = I
9020 X = 21: IF X > LN - 1 THEN X
      = LN - 1
9030 FOR I = IT + 1 TO X: PRINT
      L$(I): NEXT I
9040 EL = V1 + (IT > 21) * (IT -
      21)
9050 Q$ = "I,M:MOVE RTN:EDIT D
      ,X:DEL ESC:EXIT"
9060 VTAB 24: HTAB 1: INVERSE : PRINT
      Q$;: NORMAL
9070 POKE 34,0: POKE 35,22
9080 VTAB V1: HTAB 1: GET X$
9090 IF X$ < > "I" THEN 9130
9100 IF V1 > 1 THEN V1 = V1 - 1:
      EL = EL - 1: GOTO 9080
9110 IF EL = 1 THEN 9080
9115 EL = EL - 5: IF EL < 1 THEN
      EL = 1
9120 FOR I = EL TO EL + 20: CALL
      - 868: PRINT L$(I): NEXT : GOTO
      9080
9130 IF X$ < > "M" THEN 9180
9140 IF EL > = LN - 1 THEN 9080
9150 EL = EL + 1
9160 IF V1 < 21 THEN V1 = V1 + 1
      : GOTO 9080
9165 NN = 4: IF NN > LN - EL - 1 THEN
      NN = LN - EL - 1
9170 EL = EL + NN: VTAB 22: HTAB
      1: FOR I = EL - NN TO EL: PRINT
      L$(I): NEXT I: GOTO 9080
9180 IF X$ = CHR$( ESC) THEN 95
      90
9190 IF X$ < > "D" OR V1 = 1 THEN
      9250
9200 FOR J = EL TO LN - 1:L$(J) =
      L$(J + 1): NEXT :L$(LN) = ""
9210 X = 21 - V1: IF X > LN - EL THEN
      X = LN - EL
9220 FOR J = EL TO EL + X: CALL
      - 868: PRINT L$(J): NEXT
9230 IF EL = LN - 1 THEN V1 = V1
      - 1:EL = EL - 1
9240 LN = LN - 1: GOTO 9080
9250 IF X$ < > "X" THEN 9310
9260 VTAB 24: PRINT "DELETE FROM
      HERE TO END OF TEXT? (Y/N)
      ";: VTAB V1: HTAB 1: GET X$:
      IF X$ < > "Y" THEN 9060
9270 FOR J = EL TO LN:L$(J) = ""
      : NEXT J:LN = EL:CHAR = 1:SL
      DC = 0
9280 VTAB V1: HTAB 1: CALL - 95

```

```

8
9290 VTAB 24: INVERSE : PRINT 0$
;: NORMAL
9300 VTAB V1: HTAB 1: GOTO 9100
9310 IF X$ < > CHR$(RTN) THEN
9080
9320 L1 = EL - 8: IF L1 < 1 THEN
L1 = 1
9330 L2 = EL + 8: IF L2 > LN THEN
L2 = LN
9340 HOME : FOR J = L1 TO EL: PRINT
L$(J): NEXT J
9350 PRINT : PRINT : PRINT : PRINT
9360 FOR J = EL + 1 TO L2: PRINT
L$(J): NEXT J
9370 VTAB 24: CALL - 868: INVERSE
: PRINT " TYPE NEW LINE ABO
VE (^F TO FINISH) ";: NORMAL
9380 VTAB EL - L1 + 1: HTAB 1
9390 TLN = LN:LN = LN + 1
9400 FOR J = 1 TO 5:L$(TLN + J) =
"": NEXT J
9410 C1 = CHAR:S1 = SLOC:CHAR = 1
:SLOC = 0
9420 GOSUB 500
9430 CHAR = C1:SLOC = S1
9440 IF L$(LN) = "" THEN LN = LN
- 1
9450 NL = LN - TLN
9460 IF NL = 1 THEN 9510
9470 IF NL > 0 THEN 9500
9480 FOR J = EL TO TLN:L$(J) = L
$(J + 1): NEXT J
9490 L$(LN) = "": GOTO 9580
9500 FOR J = LN TO EL + 1 STEP -
1:L$(J + NL - 1) = L$(J): NEXT
J
9510 FOR J = 0 TO NL - 1:L$(EL +
J) = L$(LN + J):L$(LN + J) =
"": NEXT J
9520 CX$ = RIGHT$(L$(EL + NL -
1),1)
9530 IF CX$ = CR$ THEN 9580
9540 IF CX$ < > " " THEN L$(EL +
NL - 1) = L$(EL + NL - 1) +
" "
9550 SS = 1:L$ = L$(EL + NL):LL =
LEN(L$)
9560 IF MID$(L$,SS,1) < > " "
AND SS < = LL THEN SS = SS
+ 1: GOTO 9560
9570 IF LEN(L$(EL + NL - 1)) +
SS < = LWID THEN GOSUB 800
0: GOTO 9590
9580 LN = TLN + NL - 1
9590 RETURN

```

Error-handling routine. (Line 20090 is changed.)

```

19999 REM ERROR-HANDLING RTN
20000 E = PEEK(222): PRINT CHR$(
7)
20010 IF E = 0 OR E > 15 THEN PRINT
"ERROR #";E;" IN LINE "; PEEK
(218) + PEEK(219) * 256: STOP
20020 PRINT D$;"CLOSE"
20030 PRINT "DISK ERR: ";
20040 IF E = 4 THEN PRINT "WRIT
E-PROTECTED";
20050 IF E = 5 THEN PRINT "END
OF DATA";
20060 IF E = 9 THEN PRINT "DISK
FULL";
20070 IF E = 10 THEN PRINT "FIL
E LOCKED";
20080 PRINT "; PRESS A KEY";: GET
X$
20090 POKE 34,0: POKE 35,22: GOTO
210

```

ATARI®

```

*****
$      ATARI BASIC      $
$      'MICROTEXT 1.2'  $
$      AUTHOR: JON R. VOSKUIL  $
$      (C) 1982  SOFTSIDE  $
*****

```

Title page. (Lines 20 and 40 are changed.)

```

10 DIM CL$(1):CL$=CHR$(125):PRINT CL$
15 POKE 752,1
20 POSITION 8,8:PRINT "M I C R O T E X
T 1.2"
30 POSITION 12,12:PRINT "BY JON R. VOS
KUIL"
40 PRINT :PRINT "COPYRIGHT 1982 SOFTS
IDE PUBLICATIONS"
50 FOR Z=1 TO 1000:NEXT Z

```

Initialization. (Lines 120, 125, and 200 are changed.)

```

99 REM INITIALIZATION
100 PRINT CL$:TRAP 20000
120 DIM L$(40),T$(14000),B$(1),C$(1),C
R$(1),LN$(37),LNXT$(40),F$(14),S$(37)
,FT$(14),LP(500),TT$(40),X$(5)
125 DIM P$(255),PP$(80),Q$(37),F1$(14)
130 BKSP=126:RTN=155:SPC=32:B$=CHR$(12
6):CR$=CHR$(20)
135 T$="":L$="":LP(0)=0
140 CHAR=1
150 LN=1

```

```

160 LIN$="-----"
-----":S$="
"
170 OPEN #1,4,0,"K:"
180 LWID=36
185 POKE 752,0
190 V=2:H=2
200 POSITION 2,0:PRINT "^SAVE ^LOAD
^REVIEW ^EDIT ^PRINT";
210 POSITION 2,1:PRINT LIN$:
220 POSITION H,V:PRINT " ";B$;

```

Input loop.

```

499 REM INPUT LOOP
500 GET #1,C:C$=CHR$(C)
520 IF C=RTN THEN C$=CR$
639 REM BACKSPACE
640 IF C<>BKSP THEN 720
650 IF CHAR<2 THEN L$="":GOTO 500
660 PRINT B$;" ";B$;
670 CHAR=CHAR-1
675 IF CHAR<2 THEN 500
680 L$=L$(1,CHAR-1)
700 GOTO 500
719 REM CHK FOR CTRL CHAR
720 IF C<SPC THEN 2000
739 REM CHK FOR END OF LN
740 CHAR=CHAR+1:IF CHAR<LWID OR C=SPC
OR C=RTN THEN 760
750 GOSUB 1000
759 REM ADD TO STRING
760 L$(CHAR-1)=C$
779 REM RETURN
780 IF C<>RTN THEN 880
820 PRINT C$
840 GOSUB 6000:SLOC=0
850 CHAR=1
860 GOTO 500
879 REM PRINT ON SCRIN
880 PRINT C$;
899 REM UPDATE SPC PNTER, CHK FOR END
OF LINE
900 IF C=32 THEN SLOC=CHAR-1:IF CHAR=L
WID THEN GOSUB 6000:CHAR=1:SLOC=0:PRIN
T
920 GOTO 500

```

Subroutine to break line at a space and to initialize the next line.

```

1000 IF SLOC=0 THEN PRINT :GOTO 1100
1020 SS=LWID-SLOC-1:FOR J=1 TO SS:PRIN
T B$;:NEXT J
1040 FOR J=1 TO SS:PRINT " ";:NEXT J:P
RINT
1050 IF SLOC=LEN(L$) THEN LNXT$="":GOT
O 1100
1060 LNXT$=L$(SLOC+1)
1080 L$=L$(1,SLOC)

```

```

1100 GOSUB 6000
1110 L$=LNXT$
1115 LNXT$=""
1120 PRINT L$;
1140 CHAR=LEN(L$)+2
1150 SLOC=0
1160 RETURN

```

Subroutine to process command codes. (Lines 2080, 2085, 2090, and 2600 are new.)

```

1999 REM PROCESS CTRL CHAR
2000 V=PEEK(84);H=PEEK(85)
2050 TL=LEN(T$)
2060 T$(TL+1)=L$
2070 POSITION 2,0:PRINT S$:PRINT LIN$;
2080 IF C<>6 THEN 2100:REM CTRL-F (BACK TO EDIT)
2085 IF CHAR>1 THEN LP(LN)=LEN(T$);LN=LN+1;LP(LN)=LEN(T$);L$=""
2090 RETURN
2100 IF C=18 THEN GOSUB 3000:REM CTL-R

```

```

2200 IF C=19 THEN GOSUB 4000:REM CTL-S
2300 IF C=12 THEN GOSUB 5000:REM CTL-L
2400 IF C=17 THEN END :REM CTL-Q
2500 IF C=16 THEN GOSUB 7000:REM CTL-P
2600 IF C=5 AND LN>1 THEN I=LN-1;VV=V:POSITION 2,VV-1:GOSUB 9000:GOSUB 3000:REM CTL-E
2900 IF TL>0 THEN T$=T$(1,TL)
2950 GOTO 200

```

Subroutine to review entered text. (Lines 3100 and 3125 are changed; and original lines 3130, 3140, and 3150 are deleted. Line 3055 may be optionally deleted.)

```

2999 REM REVIEW TEXT
3000 PRINT CL$;"Press any key to pause";PRINT LIN$
3040 IF LN=1 THEN 3210
3050 FOR I=1 TO LN-1
3055 FOR Z=1 TO 20:NEXT Z
3060 PRINT T$(LP(I-1)+1,LP(I))

```

```

3070 IF PEEK(764)=255 AND NOT STP THE N 3200
3080 STP=0:POKE 764,255
3090 VV=PEEK(84)
3100 POSITION 2,1:PRINT LIN$;:POSITION 2,0:PRINT " RTN:Cont SPC:Stp E:Edit ";
3120 X=PEEK(764):POKE 764,255
3125 IF X=42 THEN GOSUB 9000:GOTO 3000
3160 IF X=12 THEN 3190
3170 IF X<>33 THEN 3120
3180 STP=1
3190 POSITION 2,VV:POKE 764,255
3200 NEXT I
3210 X=LP(LN-1)+1:IF X<=LEN(T$) THEN PRINT T$(X);
3220 H=PEEK(85):V=PEEK(84)
3230 RETURN

```

Subroutine to save text to disk or tape. (Lines 4075, 4082, 4083, and 4085 are changed.)

Bugs, Worms and other undesirables



The *SoftSide* Annual April Fools Bug Report (Seriously... the joke is on us!)

Add the following line to the ATARI® version of *Space Rescue* (February, 1982):

```
65 IF STRIG(0)=0 THEN B0
```

And change these lines to correct those listed in the magazine and supplied on media:

```

690 FOR I=P2 TO 1 STEP -1:SOUND 0,I*20+40,10,10:FOR W=1 TO 50:NEXT W:NEXT I:
SOUND 0,0,0,0:IF P2<6 THEN 699
693 POSITION 10,16:?"*** BONUS SHIP!
***";:SL=SL+1:FOR P2=1 TO 4:FOR AZ=80
TO 185 STEP 6
696 SOUND 0,AZ,10,10:SOUND 0,AZ+50,10,
10:NEXT AZ:NEXT P2
699 FOR W=11 TO 111 STEP 5:FOR AZ=W TO
W-7 STEP -1:SOUND 0,AZ,10,10:NEXT AZ:
NEXT W:SOUND 0,0,0,0:GOTO 20

```

In the TRS-80® version of *Space Rescue*, change the following line:

```
1000 Z=0:FORX=1TO158:READY:Z=Z+Y:NEXT:IF
Z<>15204THENCLS:PRINT"DATA BASE ERROR IN
LINES 1030-1080, CHECK LISTING.":PRINT:
```



```
LIST 1030-1080ELSEY=86:X=255:POKE-1,0:IF
PEEK(-1)<>0THENX=191:POKE-16385,0:IFPEEK
(-16385)<>0THENX=127
```

Line 6000 in COMMANDing BASIC (November, 1981) should be changed as follows:

```
60000 Z=0:FORX=1TO158:READY:Z=Z+Y:NEXT:I
FZ<>15204THENCLS:PRINT"DATA BASE ERROR I
N LINES 60060-60160, CHECK LISTING.":PRI
NT:LIST60060-60160ELSEY=86:X=255:POKE-1,
0:IFPEEK(-1)<>0THENX=191:POKE-16385,0:IF
PEEK(-16385)<>0THENX=127
```

In *Titan* ATARI® version (December, 1981) the following line should be changed:

```
8590 SETCOLOR 4,5,0:SETCOLOR 2,5,0:POS
ITION 4,22:?"The supervisor of";A$(Y*
10-9,Y*10);"has won":GOTO B640
```

TRS-80® Version

```
8590 PRINT@960,"The supervisor of";MID$(
A$,Y*10-9,10);" has won the right to min
e TITAN!";:GOTOB700
```

The ATARI® version of *Gambler* (January, 1982) should have a line added to zero the L array, as follows:

```
17 FOR I=1 TO 6:L(I)=0:NEXT I
```

Without this line it's possible to get an Error 18 under certain conditions while playing the Lottery.

Also in the ATARI® version of *Gambler*, replace line 751 with the following line. These instructions will help correct the shape of the horse in the Horse Race routine.

```
751 REM 1st routine: <SHIFT> - ' <CT
RL> <ESC><CTRL>= 5<ESC><CTRL>+ <CTRL>
' U'
```

In the Apple Program, *Rubicube* (February, 1982), there is a minor error in the instructions printed by the program. Line 2860 should be corrected as follows:

```
2860 HOME : PRINT "EACH FACE H
AS A ONE-LETTER NAME----F
RONT, P)OSTERIOR, L)EFT, R)
IGHT, T)OP, AND B)OTTOM. HIT
ANY KEY TO GO ON.": GET
A$
```

Also, in the variable list accompanying the program, variables C, C2, and C7 should refer to the column number, not the row number.

In the ATARI® *Database* program (December, 1981) the following lines should be corrected as shown, in order to clear the screen before each formatted record is printed:

```
1220 ? CHR$(125);:DN FS GOSUB 1270,131
0:POSITION N2,22:?"PRESS RETURN (ESC
FOR MENU)";
1270 IF SB=N4 THEN PRINT CHR$(125);
```

```

3999 REM SAVE TO DISK/TAPE
4000 POSITION 2,0:PRINT S$;:POSITION 2
,0:PRINT "Save to Tape or Disk? (T/D/E
SC) ";
4020 GET #1,X
4030 IF X=27 THEN 4400
4060 IF X=84 THEN 4200
4070 IF X<>68 THEN 4000
4075 F1$=F$
4080 POSITION 2,0:PRINT S$;:POSITION 2
,0:PRINT "File Name: ";:INPUT F$
4082 IF F$="" AND F1$="" THEN 4080
4083 IF F$="" THEN F$=F1$
4085 IF F$(1,1)<>"D" THEN FT$="D":FT$(
3)=F$:F$=FT$
4090 POSITION 2,0:PRINT "Insert disk a
nd press RETURN";:GET #1,X
4100 OPEN #2,0,0,F$:GOTO 4210
4200 POSITION 2,0:PRINT "Start tape re
corder and press RETURN";:GET #1,X
4205 OPEN #2,0,0,"C:"
4210 PRINT #2;LN:PRINT #2;SLOC:PRINT #
2;CHAR
4220 FOR I=1 TO LN-1
4230 PRINT #2;T$(LP(I-1)+1,LP(I))
4240 NEXT I
4250 IF CHAR>1 THEN PRINT #2;T$(LP(LN-
1)+1)
4300 CLOSE #2
4400 RETURN

```

Subroutine to load text from disk or tape. (Lines 5075, 5082, 5083, and 5085 are changed.)

```

4999 REM LOAD FROM DISK/TAPE
5000 POSITION 2,0:PRINT S$;:POSITION 2
,0:PRINT "Load from Tape or Disk? (T/D
/ESC) ";
5020 GET #1,X
5030 IF X=27 THEN 5400
5060 IF X=84 THEN 5200
5070 IF X<>68 THEN 5000
5075 F1$=F$
5080 POSITION 2,0:PRINT S$;:POSITION 2
,0:PRINT "File Name: ";:INPUT F$
5082 IF F$="" AND F1$="" THEN 5080
5083 IF F$="" THEN F$=F1$
5085 IF F$(1,1)<>"D" THEN FT$="D":FT$(
3)=F$:F$=FT$
5090 POSITION 2,0:PRINT "Insert disk a
nd press RETURN";:GET #1,X
5100 OPEN #2,4,0,F$:GOTO 5210
5200 POSITION 2,0:PRINT "Start tape re
corder and press RETURN";:GET #1,X
5205 OPEN #2,4,0,"C:"
5210 INPUT #2;LN:INPUT #2;SLOC:INPUT #
2;CHAR
5215 T$=""

```

```

5220 FOR I=1 TO LN-1
5230 INPUT #2,TT$:T$(LEN(T$)+1)=TT$
5240 LP(I)=LEN(T$)
5250 NEXT I
5255 TL=LEN(T$):L$=""
5260 IF CHAR>1 THEN INPUT #2,TT$:T$(LE
N(T$)+1)=TT$:LP(LN)=LEN(T$):L$=TT$
5300 CLOSE #2
5350 GOSUB 3000
5400 RETURN

```

Subroutine to add a line of text to the main text string.

```

5999 REM ADD LN TO TEXT STRING
6000 T$(LEN(T$)+1)=L$
6080 LP(LN)=LEN(T$)
6100 LN=LN+1:LP(LN)=LEN(T$)
6150 L$=""
6200 RETURN

```

Subroutine to print the text file in memory to a printer.

```

6999 REM PRINTOUT RTN
7000 PRINT CL$:POSITION 2,6:LIN=0
7010 ? "Left margin? (Default = 10) ";
:INPUT X$:LM=10:IF LEN(X$)>0 THEN IF V
AL(X$)>0 THEN LM=VAL(X$):IF LM>37 THEN
LM=37
7020 PRINT :PRINT "Right margin? (Defa
ult = 70) ";:INPUT X$:RM=70:IF LEN(X$)
>0 THEN IF VAL(X$)>0 THEN RM=VAL(X$)
7030 PRINT :PRINT "Line spacing? (Defa
ult = 2) ";:INPUT X$:LS=2:IF LEN(X$)>0

```

```

THEN IF VAL(X$)>0 THEN LS=VAL(X$)
7040 LL=RM-LM
7070 PRINT CL$:LPRINT "":P$="":CR=0:I=
0
7080 I=I+1:P$(LEN(P$)+1)=T$(LP(I-1)+1,
LP(I))
7090 IF P$(LEN(P$))=CR$ THEN CR=1:GOTO
7110
7100 IF LEN(P$)<255-LWID AND I<LN-1 TH
EN 7080
7110 GOSUB 7500:CR=0
7120 IF I<LN-1 THEN 7080
7130 LPRINT S$(1,LM);L$;
7150 LPRINT ""
7160 GOSUB 3000
7170 RETURN
7500 L=LL
7510 IF LEN(P$)>LL THEN 7550
7520 IF ( NOT CR) THEN 7640
7530 LP=LEN(P$):IF LP<2 THEN PP$="":P$
="":GOTO 7590
7540 PP$=P$(1,LP-1):P$="":GOTO 7590
7550 C$=P$(L,L):IF C$=" " THEN 7580
7560 L=L-1:IF L>0 THEN 7550
7570 L=LL
7580 PP$=P$(1,L):P$=P$(L+1)
7590 LPRINT S$(1,LM);PP$;
7610 FOR J=1 TO LS:LIN=LIN+1:LPRINT ""
:NEXT J
7615 IF LIN>59 THEN FOR J=1 TO 66-LIN:
LPRINT "":NEXT J:LIN=0
7620 IF LEN(P$)>LL THEN L=LL:GOTO 7550
7630 IF CR AND LEN(P$)>0 THEN 7530
7640 RETURN

```



"THE TERM IS THERAPY, MR. KRAVETS, NOT DEBUGGING."

Subroutine to readjust lines in memory so that they fit properly on the screen after editing. (This routine is wholly new.)

```

7999 REM TEXT REJUSTIFY RTN
8000 PRINT CL$:POSITION 2,5:PRINT "Re-
justifying text. . ."
8010 TL=LEN(T$):N=EL+NL-1:C=LP(N)+1:SL
=C-1:CH=C-LP(N-1)
8020 C#=T$(C,C)
8030 IF C#=CR$ THEN 8100
8035 IF C=TL THEN FL=1:SLOC=SL-LP(N-1)
:GOTO 8100
8040 IF C#=" " THEN SL=C
8050 IF CH<LWID-1 THEN CH=CH+1:C=C+1:G
OTO 8020
8060 IF SL=LP(N-1) THEN SL=C
8070 LP(N)=SL:C=SL+1:N=N+1:CH=1:GOTO 8
020
8100 LP(N)=C
8110 IF LP(N)=LP(N+1) THEN LN=LN-1:FOR
I=N TO LN-1:LP(I)=LP(I+1):NEXT I
8120 RETURN

```

Subroutine to edit lines of text. (This routine is wholly new.)

```

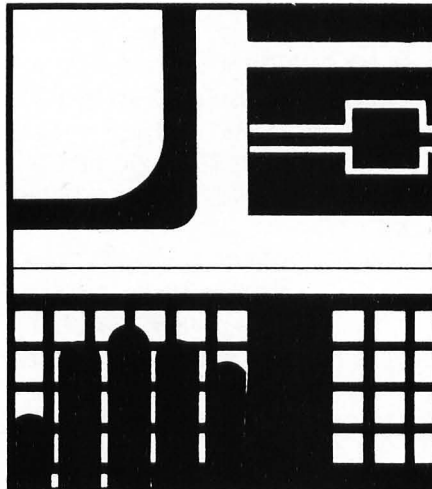
8999 REM EDIT SUBR
9000 FL=0:IF CHAR>1 THEN LP(LN)=LEN(T$
):LN=LN+1:LP(LN)=LEN(T$):L$="":FL=1
9005 POKE 752,1:IT=I:IF I>21 THEN V1=2
2:GOTO 9040
9010 V1=I+1:POSITION 2,VV
9020 X=21:IF X>LN-1 THEN X=LN-1
9025 IF X=IT THEN 9040
9030 FOR I=IT+1 TO X:PRINT T$(LP(I-1)+
1,LP(I)):NEXT I
9040 EL=V1+(IT>21)*(IT-21)-1
9050 Q$="UP/DN:Move RTN:Edit D,X:Del
ESC:Exit"
9060 POSITION 2,0:PRINT Q$:PRINT LIN#
9080 C=ASC(T$(LP(EL-1)+1)):POSITION 2,
V1:PRINT CHR$(C+128):GET #1,X
9085 POSITION 2,V1:PRINT CHR$(C);
9090 IF X<>45 THEN 9130:REM UP
9100 IF V1>2 THEN V1=V1-1:EL=EL-1:GOTO
9080
9110 IF EL=1 THEN 9080
9115 EL=EL-5:IF EL<1 THEN EL=1
9118 NN=20:IF EL+NN>LN-1 THEN NN=LN-EL
-1
9120 POSITION 2,2:FOR I=EL TO EL+NN:
S$="":T$(LP(I-1)+1,LP(I)):NEXT I:GOTO
9080
9130 IF X<>61 THEN 9180:REM DOWN
9140 IF EL>LN-1-FL THEN 9080

```

```

9150 EL=EL+1
9160 IF V1<22 THEN V1=V1+1:GOTO 9080
9165 NN=4:IF NN>LN-EL-1-FL THEN NN=LN-
EL-1-FL
9170 EL=EL+NN:POSITION 2,23:FOR I=EL-N
N TO EL:PRINT T$(LP(I-1)+1,LP(I)):NEXT
I:GOTO 9080
9180 IF X=27 THEN 9580:REM ESC
9190 IF X<>68 OR V1=2 THEN 9250:REM D
9200 NC=LP(EL)-LP(EL-1):IF EL=LN-1 THE
N T#=T$(1,LP(EL-1)):GOTO 9205
9202 T$(LP(EL-1)+1)=T$(LP(EL)+1)
9205 FOR J=EL TO LN-1:LP(J)=LP(J+1)-NC
:NEXT J
9210 X=22-V1:IF X>LN-EL-2 THEN X=LN-EL
-2
9220 POSITION 2,V1:IF EL<LN-1 THEN FOR
J=EL TO EL+X:PRINT S$;"":T$(LP(J-1)+
1,LP(J)):NEXT J

```



```

9225 PRINT S$;
9230 IF EL=LN-1-FL THEN V1=V1-1:EL=EL-
1
9240 LN=LN-1:GOTO 9080
9250 IF X<>88 THEN 9310:REM X
9260 POSITION 2,0:PRINT "Delete from h
ere to the end of text?":GET #1,X:IF
X<>89 THEN 9060
9270 LN=EL:L$="":CHAR=1:SLOC=0:IF LN>1
THEN T#=T$(1,LP(LN-1)):GOTO 9280
9275 T$=""
9280 TL=LEN(T$):GOTO 9580
9310 IF X<>155 THEN 9080
9319 REM EDIT LINE
9320 L1=EL-8:IF L1<1 THEN L1=1
9330 L2=EL+8:IF L2>LN-1 THEN L2=LN-1
9340 PRINT CL$:"Type new line below (^
F to finish) ":PRINT LIN#

```

```

9350 FOR J=L1 TO EL:PRINT T$(LP(J-1)+1
,LP(J)):NEXT J
9360 PRINT :PRINT :PRINT :PRINT
9370 IF L2>EL THEN FOR J=EL+1 TO L2:PR
INT T$(LP(J-1)+1,LP(J)):NEXT J
9380 POKE 752,0:POSITION 2,EL-L1+2
9390 TLN=LN
9410 C1=CHAR:S1=SLOC:CHAR=1:SLOC=0
9420 GOSUB 500
9430 CHAR=C1:SLOC=S1
9450 NL=LN-TLN:NC=LP(LN-1)-LP(TLN-1)
9480 IF EL=LN-1 THEN T#=T$(1,LP(EL-1))
:GOTO 9490
9485 T$(LP(EL-1)+1)=T$(LP(EL)+1)
9490 CC=LP(EL)-LP(EL-1):FOR J=EL TO LN
-1:LP(J)=LP(J+1)-CC:NEXT J:IF NL=0 THE
N LN=TLN-1:GOTO 9580
9500 X#=T$(LEN(T$)):IF X#<>CR$ AND X#<
>" " THEN T$(LEN(T$)+1)=" ":NC=NC+1:LP
(LN-2)=LP(LN-2)+1:LP(LN-1)=LP(LN-1)+1
9502 CN=LP(EL-1):FOR I=LEN(T$) TO CN+1
STEP -NC:IF I<NC THEN T$(CN+NC+1,I+NC
)=T$(CN+1,I):GOTO 9504
9503 T$(I+1,I+NC)=T$(I-NC+1,I)
9504 NEXT I
9505 T$(CN+1,CN+NC)=T$(LEN(T$)-NC+1)
9506 FOR I=LN-2 TO EL STEP -1:LP(I+NL)
=LP(I)+NC:NEXT I
9508 J=TLN+NL-1:K=LP(LN-2)-LP(EL-1):FO
R I=0 TO NL-1:LP(EL+I)=LP(J+I)-K:NEXT
I
9510 T#=T$(1,LEN(T$)-NC)
9530 LN=TLN+NL-1:IF X#=CR$ OR EL=LN-NL
THEN 9580
9550 SS=1:P#=T$(LP(EL+NL-1)+1,LP(EL+NL
)):LL=LEN(P#)
9560 IF P#(SS,SS)<>" " AND SS<LL THEN
SS=SS+1:GOTO 9560
9570 IF LP(EL+NL-1)-LP(EL+NL-2)+SS<LW
ID THEN GOSUB 8000
9580 TL=LEN(T$):IF FL THEN LN=LN-1:L#=
T$(LP(LN-1)+1,LP(LN)):CHAR=LEN(L$)+1:T
L=LP(LN-1)
9600 STP=0:POKE 752,0:RETURN

```

Error-handling routine.

```

19999 REM ERROR-HANDLING RTN
20000 CLOSE #2:E=PEEK(195)
20010 POSITION 2,0:PRINT S$;:POSITION
2,0:PRINT "Error: Code ";E;"; press an
y key";
20020 GET #1,X
20030 TRAP 20000
20040 GOTO 200

```

TRS-80[®]

```
#####  
$      TRS-80 BASIC      $  
$      "MICROTEXT 1.2"  $  
$    AUTHOR: JON R. VOSKUIL  $  
$    (C) 1982  SOFTSIDE   $  
#####
```

Title page. (Lines 20 and 40 are changed.)

```
10 CLS: PRINT CHR$(23)  
20 PRINT @198, "M I C R O T E X T  1 . 2"  
30 PRINT @462, "BY JON R. VOSKUIL"  
40 PRINT @640, "(C) 1982  SOFTSIDE PUBLICATIONS"  
50 FOR Z=1 TO 1000: NEXT Z: GOTO 100
```

Mixed-case printout conversion routine.

```
59 ' LOWERCASE PRINT RTN  
60 X$="": IF PP$="" THEN 74  
62 LOK=0: LC=-1: FOR K=1 TO LEN(PP$): CC=ASC(MID$(PP$,K,1)): IF  
CC=32 THEN LOK=0  
64 IF CC>64 THEN 70  
66 IF LC=0 THEN LOK=-1  
68 LC=0: GOTO 72  
70 LPRINT CHR$(CC-32*(LC AND CC>64 AND CC<91) * -(NOT LOK));LC  
=-1  
72 NEXT  
74 RETURN
```

Initialization. (Lines 110 and 200 are changed.)

```
99 ' INITIALIZATION  
100 CLS  
110 CLEAR 28000  
115 DEFINT A-Z  
120 DIM L$(500)  
125 ON ERROR GOTO 20000  
130 BKSP=8: RTN=13: SPC=32: CLR=31: B$=CHR$(8): CR$=CHR$(140)  
140 CHAR=1  
145 CU$=CHR$(95)  
150 LN=1  
180 LWID=62  
190 P=128  
200 PRINT @0, "SAVE:CLR-S  LOAD:CLR-L  REVW:CLR-R  EDIT:CLR-E  
PRINT:CLR-P"  
210 PRINT@ 64, STRING$(63, "-");  
220 PRINT @P, "";
```

Input loop. (Lines 530 and 740 had been omitted in version 1.1, and are included here as they should be.)

```
499 ' INPUT LOOP  
500 PRINT CU$;  
505 C$=INKEY$: IF C$="" THEN 505 ELSE C=ASC(C$)  
520 IF C=RTN THEN C$=CR$  
530 IF C>90 THEN PRINT B$;"@ "; L$(LN)=L$(LN)+"@"; CHAR=CHAR+1:  
C$=CHR$(C-32): GOTO 740  
639 ' BACKSPACE  
640 IF C=BKSP THEN IF CHAR<2 THEN 505 ELSE PRINT B$;B$; CHAR=CH  
AR-1: L$(LN)=LEFT$(L$(LN), LEN(L$(LN))-1): GOTO 500
```

```
719 ' CTRL CHARACTER  
720 IF C=CLR THEN 2000  
739 ' END OF LINE  
740 CHAR=CHAR+1: IF CHAR=LWID AND C<>SPC AND C<>RTN THEN GOSUB  
1000  
759 ' ADD TO STRING  
760 L$(LN)=L$(LN)+C$  
779 ' RETURN  
780 IF C=RTN THEN PRINT B$;C$: LN=LN+1: L$(LN)="" : SL=0: CHAR=1:  
GOTO 500  
879 ' PRINT ON SCREEN  
880 PRINT B$;C$;  
899 ' UPDATE SPC POINTER, CHK END OF LINE  
900 IF C<>32 THEN 500 ELSE SL=CHAR: IF CHAR=LWID THEN LN=LN+1: L  
$(LN)="" : CHAR=1: SL=0: PRINT  
920 GOTO 500
```

Subroutine to break line at a space and to initialize the next line.

```
999 ' JUSTIFY AND INCR LINE  
1000 IF SL=0 THEN PRINT: GOTO 1100  
1020 SS=LWID-SL: FOR J=1 TO SS: PRINT B$;: NEXT  
1040 PRINT STRING$(SS,32);:PRINT  
1050 IF SL=LWID-1 THEN 1100  
1060 L$(LN+1)= RIGHT$(L$(LN), LWID-1-SL)  
1080 L$(LN)= LEFT$(L$(LN), SL-1)  
1100 LN=LN+1  
1120 PRINT L$(LN);CU$;  
1140 CHAR=LEN(L$(LN))+2  
1150 SL=0  
1160 RETURN
```

Subroutine to process command codes. (Lines 2050 and 2600 are changed.)

```
1999 ' PROCESS CTRL CHARACTERS  
2000 P=PEEK(16416) + PEEK(16417)*256 - 15360: 'CURSOR PSN  
2020 C$=INKEY$: IF C$="" THEN 2020  
2030 C=ASC(C$)  
2050 IF C=70 THEN RETURN: 'BACK TO EDIT  
2100 IF C=82 THEN GOSUB 3000: GOTO 200: 'REVW  
2200 IF C=83 THEN GOSUB 4000: GOTO 200: 'SAVE  
2300 IF C=76 THEN GOSUB 5000: GOTO 200: 'LOAD  
2400 IF C=81 THEN END: 'QUIT  
2500 IF C=80 THEN GOSUB 7000: GOTO 200: 'PRINT  
2600 IF C=69 AND LN>1 THEN I=LN-1: PP=P: GOSUB 9000: GOSUB 3000:  
GOTO 200: 'EDIT  
2800 P=P-1: GOTO 200
```

Subroutine to review entered text. (Lines 3055, 3110, and 3130 are changed. Original lines 3140, 3145, and 3150 have been deleted.)

```
2999 ' REVIEW TEXT  
3000 CLS: PRINT@ 21, "PRESS ANY KEY TO PAUSE": PRINT STRING$(63,  
"-")  
3040 IF LN=1 THEN 3210  
3050 FOR I=1 TO LN-1  
3055 FOR Z=1 TO 20: NEXT Z  
3060 PRINT L$(I)  
3070 IF INKEY$="" AND NOT STP THEN 3200  
3080 STP=0  
3090 PP=PEEK(16416) + PEEK(16417)*256 - 15360
```

```

3110 PRINT@ 0, "  ENTER: CONTINUE      SPACEBAR: STEP 1 LINE
      E:EDIT      "; STRING$(63, "-");
3120 X%=INKEY%: IF X%="" THEN 3120
3130 X=ASC(X%): IF X=69 THEN GOSUB 9000: GOTO 3000
3160 IF X=13 THEN 3190
3170 IF X<>32 THEN 3110
3180 STP=-1
3190 PRINT @PP, " ";
3200 NEXT I
3210 PRINT L$(LN);
3220 P=PEEK(16416) + PEEK(16417)*256 - 15360
3230 RETURN

```

Subroutine to save text to disk or tape. (Lines 4075, 4085, 4088 are new.)

```

3999 ' SAVE TO DISK/TAPE
4000 PRINT@64, STRING$(63, "-");: PRINT@ 0, STRING$(63, 32);: PRINT@
  0, "SAVE TO TAPE OR DISK? (T/D/CLEAR) ";
4020 X%=INKEY%: IF X%="" THEN 4020
4040 IF X%=CHR$(CLR) THEN 4220
4050 L$(0)=STR$(LN) +STR$(10000+SL*100+CHAR)
4055 PRINT@ 0, STRING$(35, 32);
4060 IF X%="T" THEN 4190
4070 IF X%<>"D" THEN 4000
4075 F1%=F%
4080 PRINT@ 0, " ";: LINEINPUT "FILE NAME: ";F%
4085 IF F%="" AND F1%="" THEN 4080
4088 IF F%="" THEN F%=F1%
4090 PRINT@ 0, "INSERT DISK AND PRESS ENTER.": PRINT STRING$(63,
  "-");
4095 IF INKEY%<>CHR$(13) THEN 4095
4100 OPEN"D",1,F%
4140 FOR I=0 TO LN
4150 PRINT#1, L$(I)
4160 NEXT I
4170 CLOSE
4180 GOTO 4220
4190 PRINT@ 0, "START RECORDER AND PRESS ENTER. ";
4195 IF INKEY%<>CHR$(13) THEN 4195
4200 FOR I=0 TO LN
4205 PRINT#-1, CHR$(34);L$(I);CHR$(34)
4210 NEXT I
4220 P=P-1: RETURN

```

Subroutine to load text from disk or tape. (Lines 5065, 5075, and 5078 are new.)

```

4999 ' LOAD FROM DISK
5000 PRINT@64, STRING$(63, "-");: PRINT@ 0, STRING$(63, 32);: PRIN
  T@ 0, "LOAD FROM TAPE OR DISK? (T/D/CLEAR) ";
5020 X%=INKEY%: IF X%="" THEN 5020
5040 IF X%=CHR$(CLR) THEN 5250
5045 PRINT@ 0, STRING$(40, 32);
5050 IF X%="T" THEN 5210
5060 IF X%<>"D" THEN 5000
5065 F1%=F%
5070 PRINT@ 0, " ";: LINEINPUT "FILE NAME: ";F%
5075 IF F%="" AND F1%="" THEN 5070
5078 IF F%="" THEN F%=F1%
5080 PRINT@ 0, "INSERT DISK AND PRESS ENTER.": PRINT STRING$(63, "
  -");
5085 IF INKEY%<>CHR$(13) THEN 5085
5090 OPEN"I",1,F%

```

```

5110 INPUT#1, L$(0)
5120 GOSUB 5290
5130 FOR I=1 TO LN
5140 LINEINPUT#1, L$(I)
5170 NEXT I
5180 CLOSE
5200 GOTO 5250
5210 PRINT@ 0, "START RECORDER AND PRESS ENTER. ";
5215 IF INKEY%<>CHR$(13) THEN 5215
5220 INPUT#-1, L$(0): GOSUB 5290
5230 FOR I=1 TO LN
5235 INPUT#-1, L$(I)
5240 NEXT I
5250 GOSUB 3000: RETURN
5290 L%=L$(0): L=LEN(L%)
5300 CHAR= VAL(RIGHT$(L%,2)): L%=LEFT$(L%,L-2)
5310 SL= VAL(RIGHT$(L%,2)): L%=LEFT$(L%,L-5)
5320 LN=VAL(L%)
5330 RETURN

```

Subroutine to print the text file in memory to a printer.

```

6999 ' PRINTOUT RTN
7000 CLS: LIN=0
7010 X%="10": INPUT"LEFT MARGIN (DEFAULT = 10) ";X%: LM=VAL(X%)
7020 PRINT: X%="70": INPUT"RIGHT MARGIN (DEFAULT = 70) ";X%: RM=
  VAL(X%)
7030 PRINT: X%="2": INPUT"LINE SPACING (DEFAULT = 2) ";X%: LS=VA
  L(X%)
7040 LL=RM-LM
7050 PRINT: X%="N": INPUT"CONVERT TO LOWERCASE, UNLESS PRECEDED
  BY @ (DEFAULT = NO) ";X%: UC=-1: IF LEFT$(X%,1)="Y" THEN UC=0
7070 CLS: LPRINT"": P%="": CR=0: I=0
7080 I=I+1: P%=P%+L$(I)
7090 IF RIGHT$(P%,1)=CR% THEN CR=-1: GOTO 7110
7100 IF LEN(P%)<255-LWID AND I<LN THEN 7080
7110 GOSUB 7500: CR=0
7120 IF I<LN THEN 7080
7130 LPRINT TAB(LM);: IF UC THEN LPRINT P%;: GOTO 7150
7140 PP%=P%: GOSUB 60
7150 LPRINT""
7160 GOSUB 3000
7170 RETURN
7500 L=LL
7510 IF LEN(P%)>LL THEN 7550
7520 IF NOT CR THEN 7640
7530 LP=LEN(P%): IF LP<2 THEN PP%="": P%="": GOTO 7590
7540 PP%=LEFT$(P%,LP-1): P%="": GOTO 7590
7550 C%=MID$(P%,L,1): IF C%=" " THEN 7580
7560 L=L-1: IF L>0 THEN 7550
7570 L=LL
7580 PP%=LEFT$(P%,L): P%=RIGHT$(P%,LEN(P%)-L)
7590 LPRINT TAB(LM);: IF UC THEN LPRINT PP%;: GOTO 7610
7600 GOSUB 60
7610 FOR J=1 TO LS: LIN=LIN+1: LPRINT"": NEXT J
7615 IF LIN>59 THEN FOR J=1 TO 66-LIN: LPRINT"": NEXT J: LIN=0
7620 IF LEN(P%)>LL THEN L=LL: GOTO 7550
7630 IF CR AND LEN(P%)>0 THEN 7530
7640 RETURN

```

Subroutine to readjust lines in memory so that they fit properly on the screen after editing. (This routine is wholly new.)


```

7999 ' TEXT REJUSTIFY RTN
8000 CLS: PRINT"RE-JUSTIFYING TEXT. . ."
8010 LIN=EL: LN=LN-1

8020 P$="": CR=0: I=EL-1
8030 I=I+1: P$=P$+L$(I)
8040 IF RIGHT$(P$,1)=CR$ THEN CR=-1: GOTO 8060
8050 IF LEN(P$)+LEN(L$(I+1))<256 AND I<LN THEN 8030
8060 GOSUB 8500
8070 IF NOT CR THEN 8100
8079 ' END OF PARAGRAPH

8080 X=I+1-LIN: IF X=0 THEN 8150
8089 ' MOVE LNS DOWN IN ARRAY
8090 FOR J=I+1 TO LN: L$(J-X)=L$(J): NEXT J: LN=LN-X: GOTO 8150
8100 IF I<LN THEN 8030
8110 L$(LIN)=P$
8120 CHAR=LEN(P$)+1: SL=LEN(P$)

8130 FOR I=LIN+1 TO LN: L$(I)="": NEXT I
8140 LN=LIN
8150 RETURN
8500 L=LWID
8510 IF LEN(P$)>LWID THEN 8550
8520 IF NOT CR THEN 8630

8530 LP=LEN(P$)
8540 PP$=LEFT$(P$,LP): P$="": GOTO 8590
8550 C$=MID$(P$,L,1): IF C$=" " THEN 8580
8560 L=L-1: IF L>0 THEN 8550
8570 L=LWID

8580 PP$=LEFT$(P$,L): P$=RIGHT$(P$,LEN(P$)-L)
8590 L$(LIN)=PP$
8600 LIN=LIN+1
8610 IF LEN(P$)>LWID THEN L=LWID: GOTO 8550
8620 IF CR AND LEN(P$)>0 THEN 8530
8630 RETURN

```

Subroutine to edit lines of text. (This routine is wholly new.)

```

8999 ' EDIT LN SUBRTN
9000 IT=I: IF I>13 THEN V1=15: GOTO 9040
9010 V1=I+2: PRINT @PP,;
9020 X=13: IF X>LN-1 THEN X=LN-1
9030 FOR I=IT+1 TO X: PRINT L$(I): NEXT I
9040 EL=V1-(IT>13)*-(IT-13)-2

9050 Q$=" UP/DOWN ARROWS:MOVE ENTER:EDIT D,X:DELETE
CLR:EXIT"
9060 PRINT @0,Q$: PRINT STRING$(63,"-");
9080 PRINT @V1#64-2,"<E";
9082 IF PEEK(14400)>0 THEN 9088
9085 X$=INKEY$: IF X$="" THEN 9085
9088 PRINT B$;B$;
9090 IF X$<>CHR$(91) THEN 9130
9100 IF V1>3 THEN V1=V1-1: EL=EL-1: GOTO 9080
9110 IF EL=1 THEN 9080
9115 EL=EL-5: IF EL<1 THEN EL=1

9120 PRINT @127,"": FOR I=EL TO EL+12: PRINT L$(I): NEXT: GOTO 9
080
9130 IF X$<>CHR$(10) THEN 9180
9140 IF EL>LN-1 THEN 9080
9150 EL=EL+1
9160 IF V1<15 THEN V1=V1+1: GOTO 9080

```

```

9165 NN=4: IF NN>LN-EL-1 THEN NN=LN-EL-1
9170 EL=EL+NN: PRINT @960,;: FOR I=EL-NN TO EL: PRINT L$(I): NEX
T I: GOTO 9060
9180 IF X$=CHR$(CLR) THEN 9590
9190 IF X$<>"D" OR V1=3 THEN 9250
9200 FOR J=EL TO LN-1: L$(J)=L$(J+1): NEXT: L$(LN)=" "
9210 X=15-V1: IF X>LN-EL THEN X=LN-EL
9220 PRINT @V1#64-65,"": FOR J=EL TO EL+X: PRINT L$(J): NEXT
9230 IF EL=LN-1 THEN V1=V1-1: EL=EL-1
9240 LN=LN-1: GOTO 9080
9250 IF X$<>"X" THEN 9310
9260 PRINT @0, "DO YOU WANT TO DELETE FROM HERE TO THE END OF TH
E TEXT? (Y/N) ";
9262 PRINT@V1#64-2,"<E";

9265 X$=INKEY$: IF X$="" THEN PRINT@V1#64-2," ";GOTO 9262
9268 IF X$<>"Y" THEN 9060
9270 FOR J=EL TO LN: L$(J)="": NEXT J: LN=EL: CHAR=1: SL=0
9280 PRINT @V1#64-65,"": FOR J=V1 TO 14: PRINT: NEXT J
9290 PRINT @0, Q$;

9300 PRINT @V1#64-66,"<E";: GOTO 9100
9310 IF X$<>CHR$(RTN) THEN 9080
9320 L1=EL-4: IF L1<1 THEN L1=1
9330 L2=EL+4: IF L2>LN THEN L2=LN
9340 CLS: PRINT"TYPE NEW LINE BELOW (CLR-F TO FINISH)": PRINT ST
RING$(63,"-")

9350 FOR J=L1 TO EL: PRINT L$(J): NEXT J
9360 PRINT: PRINT: PRINT: PRINT
9370 FOR J=EL+1 TO L2: PRINT L$(J): NEXT J
9380 PRINT @(EL-L1+2)#64,;
9390 TLN=LN: LN=LN+1

9400 FOR J=1 TO 5: L$(TLN+J)="": NEXT J
9410 C1=CHAR: S1=SL: CHAR=1: SL=0
9420 GOSUB 500
9430 CHAR=C1: SL=S1
9440 IF L$(LN)=" " THEN LN=LN-1

9450 NL=LN-TLN
9460 IF NL=1 THEN 9510
9470 IF NL>0 THEN 9500
9480 FOR J=EL TO TLN: L$(J)=L$(J+1): NEXT J
9490 L$(LN)="": GOTO 9580

9500 FOR J=LN TO EL+1 STEP-1: L$(J+NL-1)=L$(J): NEXT J
9510 FOR J=0 TO NL-1: L$(EL+J)=L$(LN+J): L$(LN+J)="": NEXT J
9520 CX$=RIGHT$(L$(EL+NL-1),1)
9530 IF CX$=CR$ THEN 9580
9540 IF CX$<>" " THEN L$(EL+NL-1)=L$(EL+NL-1)+" "

9550 SS=1: L$=L$(EL+NL): LL=LEN(L$)
9560 IF MID$(L$,SS,1)<>" " AND SS<LL THEN SS=SS+1: GOTO 9560
9570 IF LEN(L$(EL+NL-1))+SS <=LWID THEN GOSUB 8000: GOTO 9590
9580 LN=TLN+NL-1
9590 RETURN

```

Error-handling routine.

```

19999 ' ERROR-HANDLING RTN
20000 PRINT@0,STRING$(63,32);
20010 E=ERR/2+1: PRINT@0, "ERROR: CODE";E;
20050 PRINT"; PRESS ANY KEY";
20060 IF INKEY$="" THEN 20060
20070 P=P-1: RESUME 200

```

FIRST: MONSTER MOVIES NOW:
CRUSH, CRUMBLE & CHOMP!
THE GREAT MOVIE MONSTER COMPUTER GAME!

And guess who stars as the movie monster. You! As any of six different monsters. More if you have the disk version.

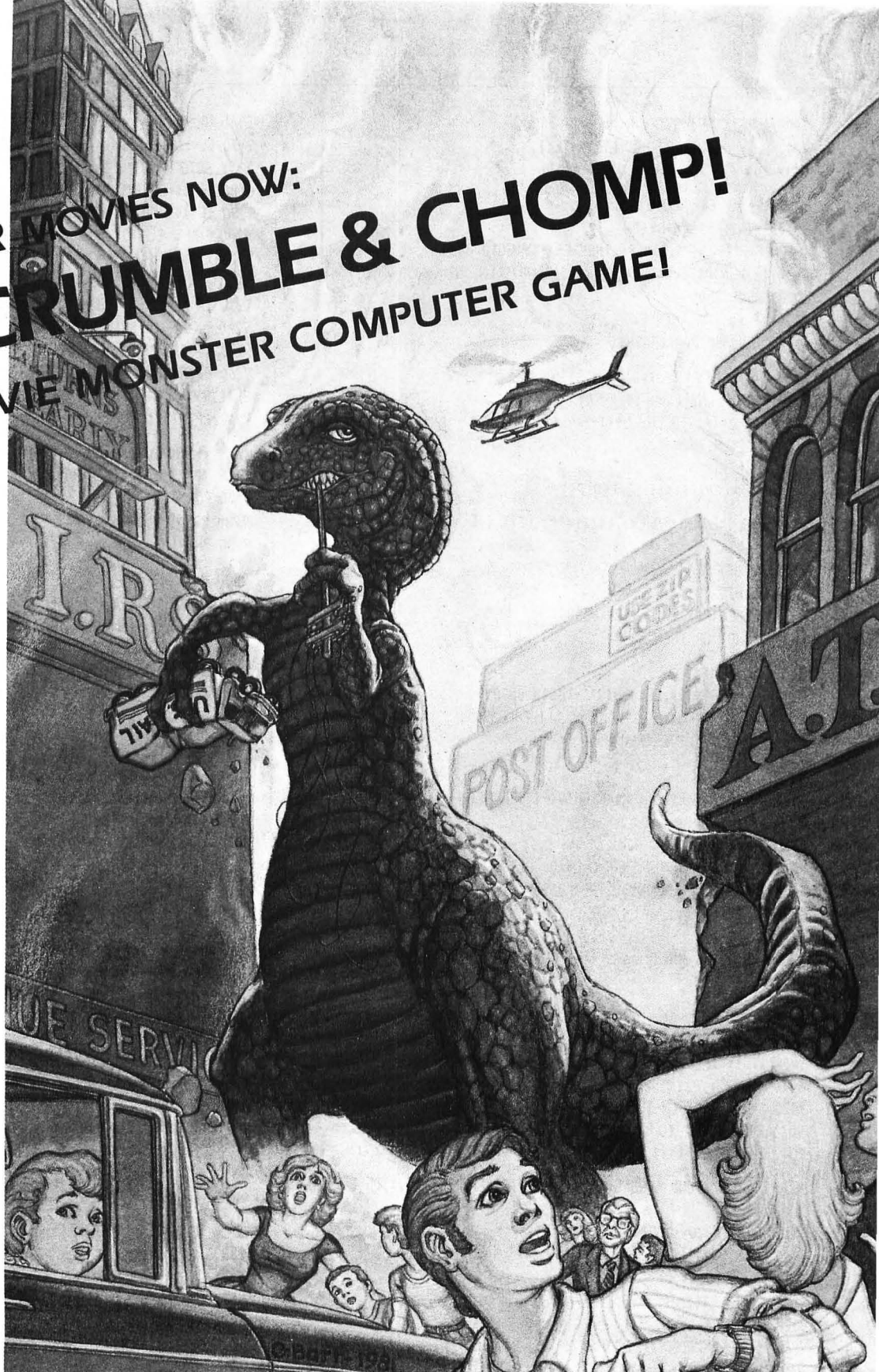
You can terrorize and destroy four of the world's largest and most densely populated cities in over 100 possible scenarios. From Tokyo to the Golden Gate, you are the deadliest creature in the air, on the land, or in the sea.

You can be the deadly amphibian who simultaneously smashes street cars, lunches on helpless humans and radiates a ray of death.

If you were a giant winged creature, think of the aerial attacks you could make on the terrified but tasty tidbits beneath you.

But as in all the best monster movies, you're up against everything the human race can throw at you—even nuclear warheads and a strange concoction developed by a team of mad scientists.

For only \$29.95 you get 6 stupendous monsters, each with its own monstrous summary card, 4 teeming metropoli displayed in graphic detail on your computer display and mapped in the accompanying 48-page illustrated book, the awesome sounds of monsterly mayhem, and spine-tingling, real-time, edge-of-your-seat excitement.



GET CRUSH, CRUMBLE & CHOMP
now at your local dealer for your APPLE, ATARI,
or TRS-80 .. **before it's too late.**




Explanation Of ATARI® Line Listings

SoftSide uses the following conventions in representing unprintable characters in ATARI® line listings, unless otherwise noted:

Characters (including blank spaces) which are underlined should be typed in inverse video.

When graphics or control characters are to be included in a string (between quotation marks), that fact will be noted in a nearby REMark. In such cases, graphics characters are represented by the corresponding lower-case letter, and control characters are represented by the corresponding unshifted key symbol. For example: The lower-case letter *s* represents a graphics cross, entered by holding down the CTRL key and then pressing the S key. The symbol = represents a control-down-arrow, entered by first pressing and releasing the ESC key, then holding down the CTRL key and pressing the = key. (See Appendix F, and the back cover, of the *ATARI® BASIC Reference Manual*.)

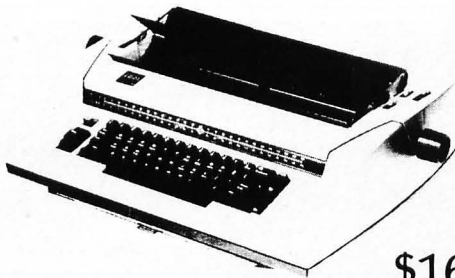
The one regular exception to the above is that a clear-screen character (ESC CTRL-<) is represented in listings by a right-hand brace, which looks like this: }

A shifted = is represented in the listings by a vertical line with a small gap in it. 

WHEN YOU SPEND SO MUCH FOR A PRINTER,
YOU SHOULD HAVE ONE THAT YOU CAN USE---

Introducing....

THE IBM TOTAL PRINTER/TYPEWRITER



\$1695

FEATURES:

10 and 12 Pitch, Proportional Space, Full Typewriter Use, Auto Correcting, Sound Cover, "Smart" Keyboard

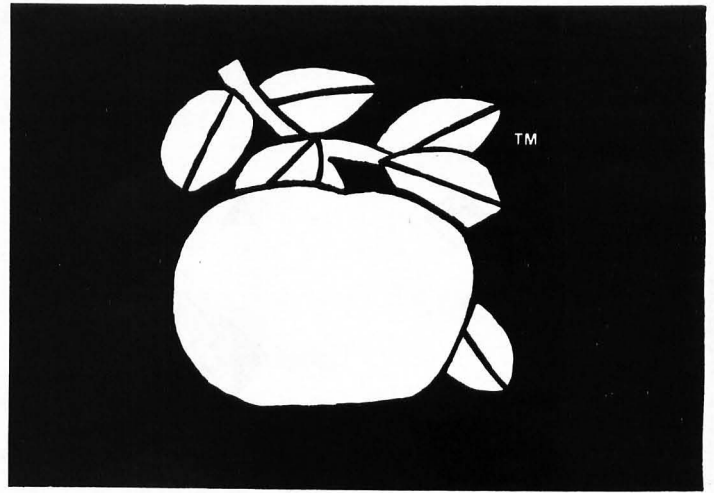
SPECIFICATIONS:

200 WPM Throughput, Either Serial OR Parellel, Self-Test, Lowest On-Site Maintenance, IBM Backed Printer. Cables stocked for all Apple, TRS(I, II, III), RS-232 systems.

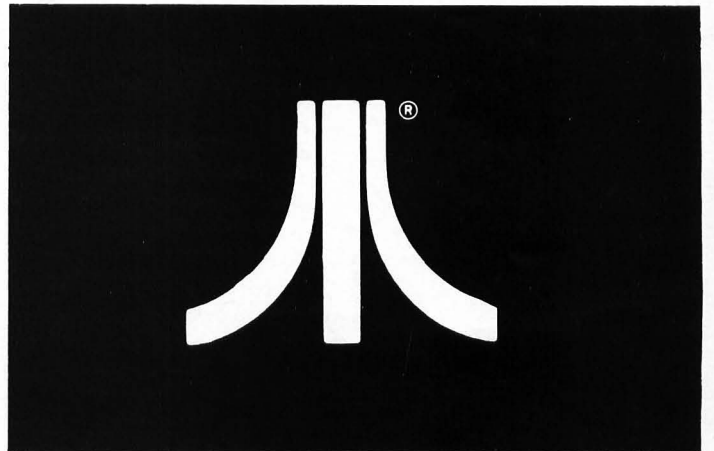
GUARANTEE: 30 days on-site by IBM

CONTACT:

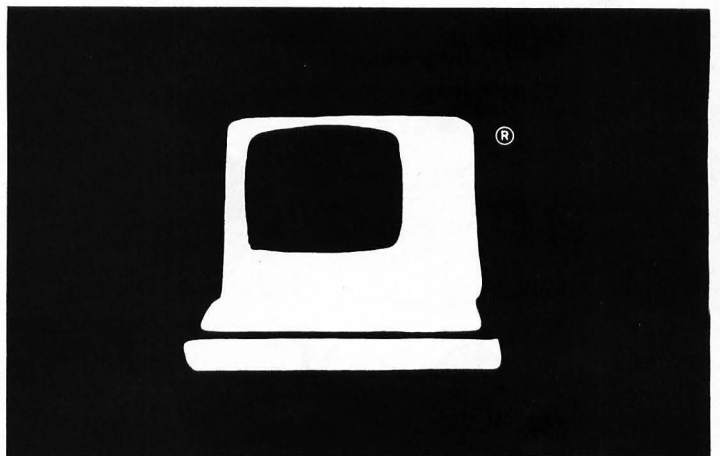
ICOM 11 N. Main, Lombard, Illinois 60148 (312) 932-1766



APPLE™/SIDE **34**
page_____

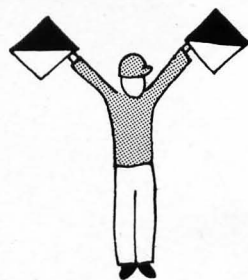


ATARI®/SIDE **54**
page_____

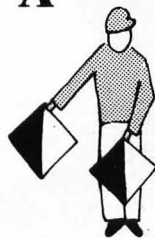


TRS-80®/SIDE **67**
page_____

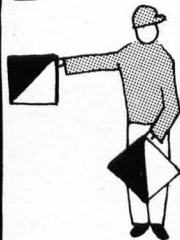
ATTENTION!



A



B



C



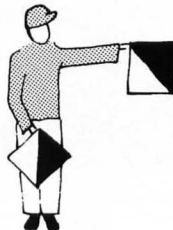
D



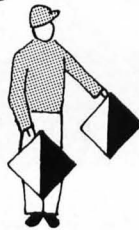
E



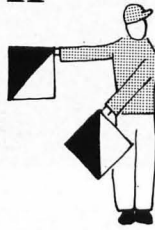
F



G



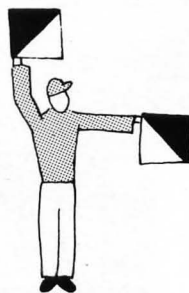
H



I



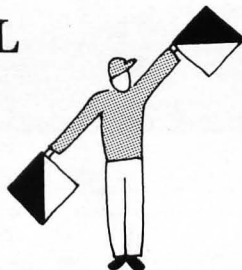
J



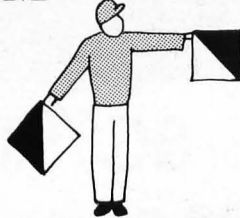
K



L



M



SEMAPHORE

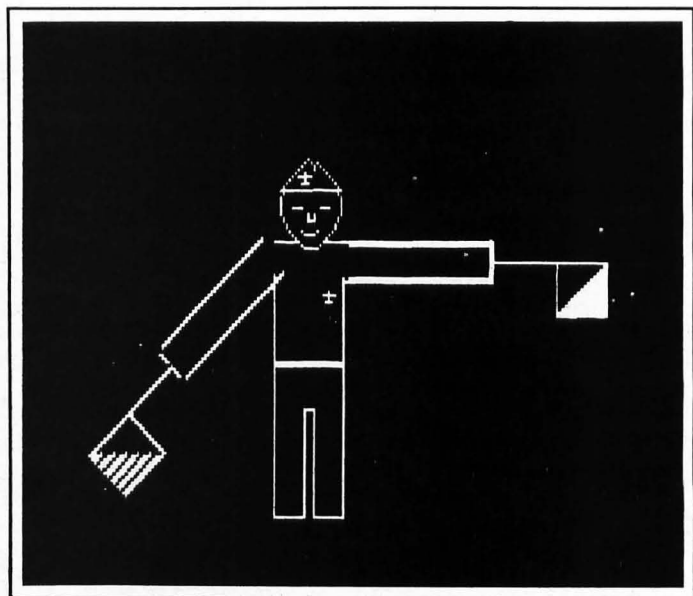
by Richard A. Bryant

Semaphore is a Hi-Res graphic program for an Apple II with Applesoft, 48K RAM, and disk drive. It is included as a bonus program on this month's Apple Disk Version.

Once in a while *SoftSide* receives a program submission which is truly unique. This program, written by a veteran Boy Scouts of America Scoutmaster, demonstrates the use of semaphore flags on the Apple's Hi-Res graphics screen.

It offers both a teaching and a testing mode. In the teaching mode, the message being sent appears on the screen as it is being sent, with a cursor advancing from letter to letter. In the testing mode, the message does not appear until after it has been sent.

As the receiver of the message, you can choose whether to respond or not as it is being sent. If you choose to respond, the program will pause after the initial "ATTENTION" has been sent and after each word sent; at the end of the entire message, you can signal either that






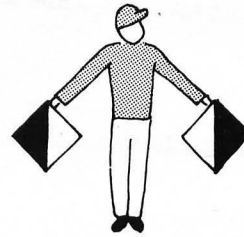
PHORIE

you have successfully received it or that you need to have it repeated.

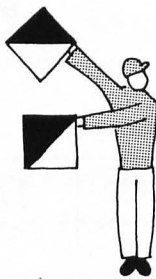
Strings of up to 79 characters can be sent at one time (one character less than two full lines). Note that numbers should be spelled out rather than entered as digits. A range of relative sending speeds is available, from 0 (slow) to 500 (fast). A good learning speed would be around 100; a good testing speed about 350. Five preset messages are built into the program for practice.

On the disk, the program is distributed in four files. The one titled SEMAPHORE is the main control program which loads the shape table (SEMAPHORE SHAPES A\$5060 L\$5A0) and runs PART 1A, which displays the main menu and instructions if desired. PART 2A is the Hi-Res program of a Boy Scout sending semaphore, and is made to print out on an Axiom IMP-2 if desired. On line 138, P equals the printer card slot, and line 192 accomplishes the actual print-out. These, along with line 1120, can be eliminated or changed if you have no suitable graphics printer or one that needs different instructions. 

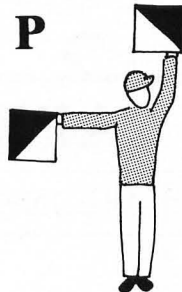
N



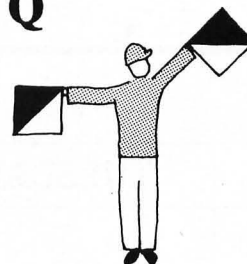
O



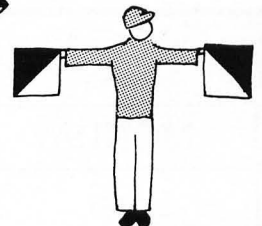
P



Q



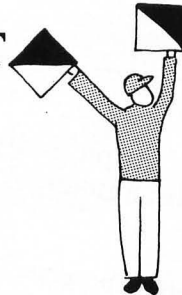
R



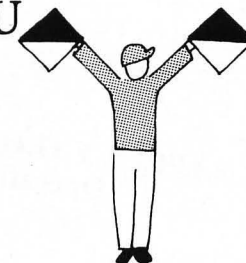
S



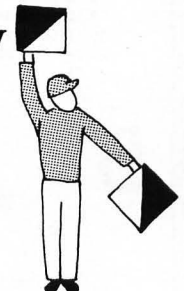
T



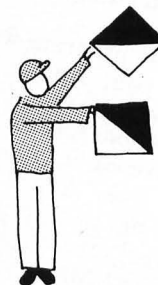
U



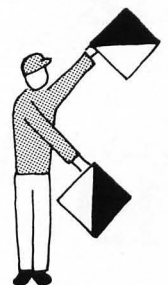
V



W



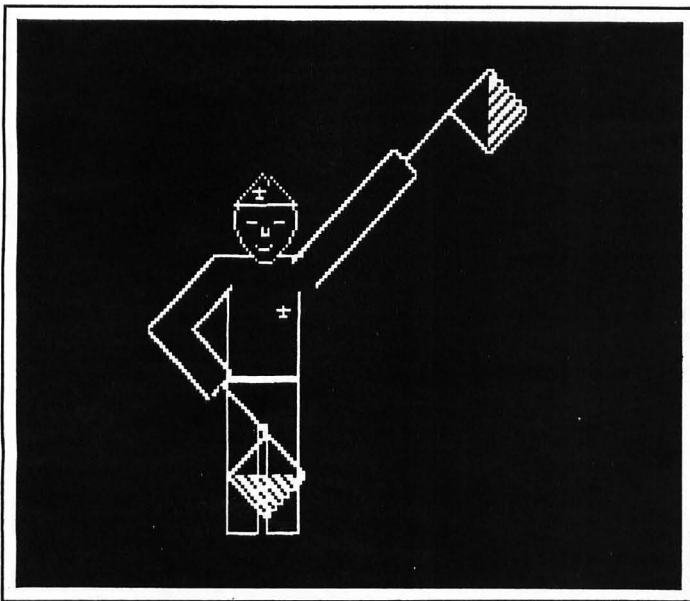
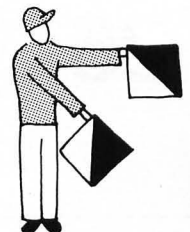
X



Y



Z



K-Byter

Battleship

An Applesoft K-Byter by David Bahr, Appleton, WI

This is a battleship program that involves more than simply trying to come up with random numbers. It requires a knowledge of direction, and the skill of deductive reasoning.

The computer begins by hiding seven ships on a 10 by 10 grid. Having done this, it will display the grid, giving no clue as to where the ships were placed. You will then be asked what coordinates you would like to bomb. After you input the numbers, you will be told if you hit a ship, and your sonar will tell you how many ships lie in each direction from your chosen coordinates.

By placing your shots in strategic positions, your sonar and your deductive powers can home in on the most likely locations for the concealed ships. You'll find it a challenge to destroy all seven ships using less than the suggested "good" score of 20 shots.

```

100 H = 7: DIM A(9,9):B$ = CHR$(
    (7) + CHR$(7))
110 FOR Z = 1 TO H
120 AX = INT ( RND (1) * 10):AY =
    INT ( RND (1) * 10)
130 IF A(AX,AY) = 1 THEN GOTO 1
    20
140 A(AX,AY) = 1: NEXT : GOTO 220

150 VTAB 19: INPUT "INPUT X,Y CO
    ORDINATES:";AX,AY
160 IF DX < 0 OR DX > 9 OR DY <
    0 OR DY > 9 THEN 150
170 T = T + 1: IF A(AX,AY) = 1 THEN
    GOTO 200
180 IF A(AX,AY) = 4 GOTO 220
190 A(AX,AY) = 3: GOTO 220
200 HOME : VTAB 12: PRINT TAB(

```

```

    15);"IT'S A HIT";B$
210 FOR Z = 1 TO 700: NEXT :H =
    H - 1:A(AX,AY) = 4
220 HOME : HTAB 14: PRINT " 012
    3456789"
230 FOR Y = 0 TO 9: HTAB 15: PRINT
    Y;: FOR X = 0 TO 9: DN A(X,Y
    ) GOTO 240,240,250,260
240 PRINT "+";: GOTO 270
250 PRINT " ";: GOTO 270
260 PRINT "*";
270 NEXT : PRINT : NEXT
280 VTAB 4: HTAB 32: PRINT "^": HTAB
    32: PRINT "'": HTAB 32: PRINT
    "N"
290 VTAB 5: HTAB 2: PRINT "SHIPS
    LEFT": HTAB 6: PRINT H
300 IF T = 0 THEN 150
310 IF H = 0 THEN 420
320 VTAB 14: PRINT "SONAR REPORT
    "
330 W = 0:E = 0:N = 0:S = 0
340 FOR Y = 0 TO 9: FOR X = 0 TO
    9
350 IF AX < 9 AND X > AX THEN IF
    A(X,Y) = 1 THEN E = E + 1
360 IF AY > 0 AND Y < AY THEN IF
    A(X,Y) = 1 THEN N = N + 1
370 IF AY < 9 AND Y > AY THEN IF
    A(X,Y) = 1 THEN S = S + 1
380 IF AX > 0 AND X < AX THEN IF
    A(X,Y) = 1 THEN W = W + 1
390 NEXT X,Y
400 PRINT " THERE ARE ";N;" SHI
    PS NORTH, ";S;" SHIPS
    SOUTH, ";E;" SHIPS EAST, AN
    D ";W;" SHIPS WEST OF COOR
    DINATES ";AX;",";AY;". "
410 GOTO 150
420 VTAB 17: HTAB 15: PRINT "YOU
    WIN": PRINT B$;B$: PRINT : PRINT
    " IT TOOK ";T;" SHOTS.": PRINT
    " (BELOW 20 IS GOOD)": END

```



WHITE LIGHTNING

by Randy Fox

White Lightning is a graphics game program for an Apple with Applesoft and 16K RAM.

This fast-moving color graphics game pits you against time and the limitations of your reflexes as you direct a moving line around the screen to intercept targets. The targets appear and disappear at random. Each time you hit one you increase your score; but at the same time, your line grows longer and becomes more difficult to maneuver. Hitting a screen border or your own "tail" (including doing an abrupt about-face) ends the game.

The constantly moving line is directed from the keyboard using the E and D keys to turn it up or down, and the O and P keys to turn it left or right. (If you prefer using other keys, simply change the four numbers in lines 26-32 of the "read keyboard" subroutine.)

When your score reaches the size of the I% array, the line breaks. The forward portion of the line con-

tinues as before, and the tail portion remains stationary as an obstacle, adding to the fun!

If you're a compulsive program tinkerer and like to make modifications, you might want to try adjusting the multiplier values in the random number generators for TS (target score) and TT (target's time on the screen), in lines 222 and 224. This can make the game quicker or more sluggish, with higher or lower scores, as you wish.

Variables

A\$: Input string.
 CA-CG: Constant values from -1 through 5.
 CH: Constant value of 15 (the color of the tail).
 DI: Number of elements in array I%.
 GC: Game counter. Used to play "You Lose" music only every fifth game.
 GS: Grand score for current game.
 HI: High score for all games played at one sitting.

HT: Total number of targets hit in the current game.

HW: "Hit what?" — color value from SCRN function.

I%: Integer array. Holds the previous moves by the leading point of the line by counter M. Decoded by counter R, the last block of the tail is erased as the front block is added.

KB: Value from the keyboard location PEEK.

LE: Game difficulty level.

OM: Value to be encoded into I%. Values 1 through 4 indicate moves up, right, down, or left.

PH: Percentage of plotted targets hit in the current game.

R, M: Counters to decode (remember) and encode (memorize) data for I%.

TA: Average points per target hit in current game.

TP: Total number of targets plotted in current game.

TT: Target's time on screen.

TX, TY: X and y coordinates of the target.

XL, YL: X and y coordinates of the screen location to be erased from the end of the tail (decoded from I%).

XM, YM: X-move and y-move values. Either may be -1, 0, or 1, to move the line right, left, up, or down.

XN, YN: New x and y coordinates of the leading end of the line.

XR, YR: X and y move values calculated from decoding I%.

Z: FOR-NEXT loop index.


```

#####
$   APPLESOFT BASIC   $
$   'WHITE LIGHTNING' $
$   AUTHOR: RANDY FOX  $
$   (C) 1982  SOFTSIDE $
#####

```

Dimension I% and then jump to title page.

```

13 TEXT : RESTORE : DIM I%(230)
15 GOTO 782

```

Read keyboard. Set x-move, y-move, and old-move (XM, YM, and OM) values.

```

22 KB = PEEK ( - 16384)
24 IF R + 2 = M THEN GOSUB 102
26 IF KB = 196 THEN XM = CB:YM =
    CC:OM = CF: RETURN
28 IF KB = 197 THEN XM = CB:YM =
    CA:OM = CE: RETURN
30 IF KB = 207 THEN XM = CA:YM =
    CB:OM = CD: RETURN
32 IF KB = 208 THEN XM = CC:YM =
    CB:OM = CC: RETURN
34 RETURN

```

Print title line.

```

42 GOSUB 92: PRINT TAB( 4) "- W
    H I T E  L I G H T N I N G
    -": PRINT : GOSUB 92: RETURN

```

Check keyboard for keypress. Utilizes two entry points: Line 52 prints a prompt string; entering at line 54 skips the prompt.

```

52 PRINT TAB( 8) "PRESS ANY KEY
    TO CONTINUE"
54 KB = PEEK ( - 16384): IF KB <
    127 THEN 54
56 POKE - 16368,0: RETURN

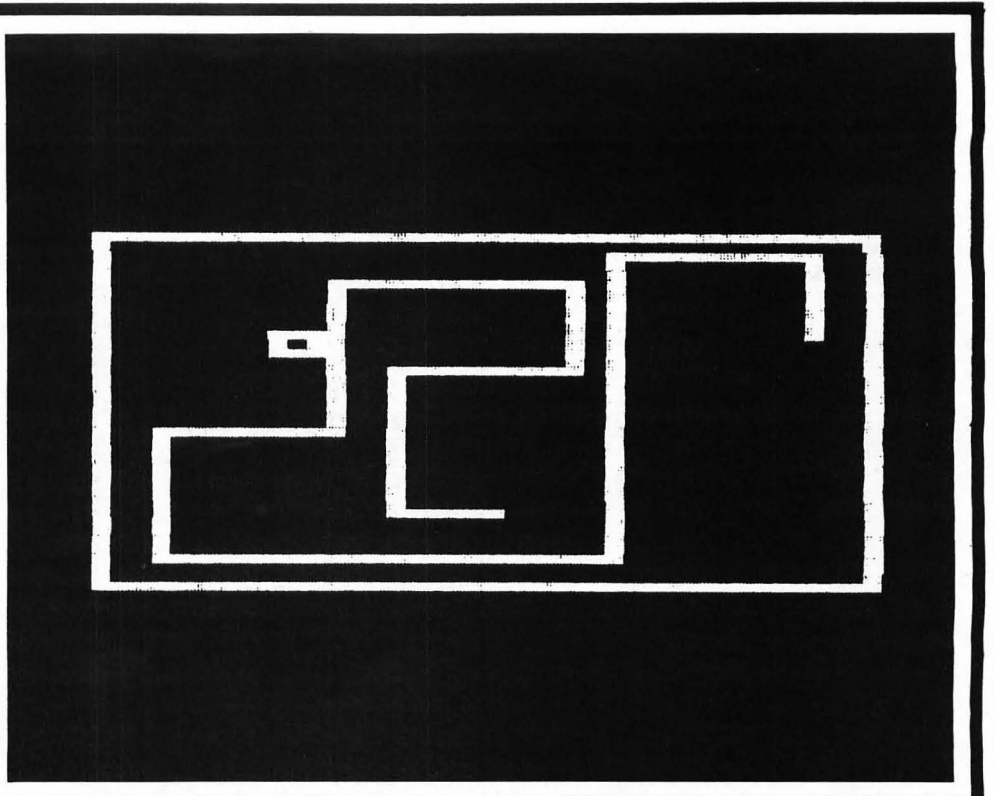
```

Initialize variables for a new game.

```

72 CA = - 1:CB = 0:CC = 1:CD = 2
    :CE = 3:CF = 4:CG = 5:CH = 1
    :OM = 1
74 TP = 0:HT = 0:L = 1:TT = 1:R =
    0:M = 0:GS = 0

```



```

76 XN = 20:YN = 20:XR = 20:YR = 2
    0:XM = 1:YM = 0:XL = 1:YL =
    0
78 DI = 230
80 RETURN

```

Draw a line of asterisks.

```

92 FOR Z = 0 TO 39: PRINT "*"; NEXT
    Z: PRINT : RETURN

```

Reset variables when tail breaks.

```

102 XR = XN:YR = YN:XL = XM:YL =
    YM:M = 0:R = 0: RETURN

```

Plot routine, to draw target. Color is determined by other subroutines.

```

162 PLOT TX + CA,TY: PLOT TX + C
    C,TY: HLINE TX + CA,TX + CC AT
    TY + CA: HLINE TX + CA,TX + C
    C AT TY + CC
164 RETURN

```

Draw playing field for higher difficulty level.

```

182 READ A,B,C
184 IF A = CH THEN 190
186 HLINE XN + A,XN + B AT YN + C
188 GOTO 182
190 READ A,B,C

```

```

192 IF A = CH THEN GOTO 198
194 VLINE YN + A,YN + B AT XN + C
196 GOTO 190
198 READ A,B
200 IF A = CH THEN RESTORE : GOTO
    390
202 PLOT XN + A,YN + B
204 GOTO 198
208 DATA -2,2,-5,-5,-2,-2,-5,-2
    ,2,-2,2,5,2,5,2,2,5,-2
210 DATA -16,-10,-1,-16,-10,1,
    10,16,1,10,16,-1,15,15,15
212 DATA -4,-3,-2,-1,1,-5,3,4,-
    2,3,4,2,-1,1,5,-3,-4,2
214 DATA -16,-10,-1,-16,-10,1,
    10,16,1,10,16,-1,15,15,15
216 DATA 0,-3,-3,0,3,0,0,3,0,0
    ,15,15

```

Random number generators for target's x and y coordinates, point value, and time on the screen (TX, TY, TS, and TT).

```

222 TS = INT ( RND ( 1) * 16): IF
    TS < 1 THEN GOTO 222
224 TT = INT ( RND ( 1) * 22) + 2
    5: IF TT < 18 THEN GOTO 224
226 TX = INT ( RND ( 1) * 37): IF
    TX < 3 THEN GOTO 226
228 TY = INT ( RND ( 1) * 37): IF
    TY < 3 THEN GOTO 228
230 VTAB 23: HTAB 30: PRINT "
    "; HTAB 30: PRINT TS

```

Search coordinates adjacent to TX, TY for screen color white. Jump to subroutine 222 to pick new coordinates if white is found.

```
242 FOR A = - 1 TO 1
244 FOR B = - 1 TO 1
246 IF SCRN( TX + A,TY + B) = C
  H THEN GOTO 226
248 IF SCRN( TX + A,TY + B) = C
  D THEN GOTO 226
250 NEXT B
252 NEXT A
```

Count number of targets printed.

```
262 TP = TP + CC
```

Set target color and draw target with sound.

```
282 COLOR= CF: POKE 768,190: POKE
  769,10: CALL 770: POKE 768,1
  90: POKE 769,10: CALL 770: POKE
  768,144: POKE 769,20: CALL 7
  70: GOSUB 162:L = CB: GOTO 5
  02
```

Flash a "hit" target and erase the target with sound.

```
302 POKE 768,113: POKE 769,10: CALL
  770
304 COLOR= CB: GOSUB 162
306 POKE 768,113: POKE 769,10: CALL
  770
308 COLOR= CF: GOSUB 162
310 POKE 768,113: POKE 769,10: CALL
  770
312 COLOR= CB: GOSUB 162
314 POKE 768,144: POKE 769,56: CALL
  770
```

Increase grand score (GS), add to tail, and check keyboard. End game if the screen border or tail is struck.

```
332 GS = GS + TS:HT = HT + CC
334 VTAB 23: HTAB 12: PRINT GS
336 R = R + CC: IF R > DI THEN R =
  CB
338 IZ(R) = OM: COLOR= CH: PLOT X
  N,YN
340 V = TS + CC
```

```
342 V = V + CA: IF V < 1 THEN GOTO
  222
344 GOSUB 22
346 R = R + CC: IF R > DI THEN R =
  CB
348 IZ(R) = OM: COLOR= CH: XN = XN
  + XM: YN = YN + YM
350 HW = SCRN( XN,YN): IF HW = C
  D OR HW = CH THEN 602
352 FOR Z = 1 TO 40: NEXT
354 PLOT XN,YN: GOTO 342
```

Draw the playing field. Add obstacles as selected by skill level variable (LE=0 denotes a normal game, LE=1 denotes increased difficulty).

```
372 HOME : GR : COLOR= CD
374 GC = GC + 1
376 FOR Z = 0 TO 39
378 PLOT Z,0: PLOT 39,Z: PLOT 39
  - Z,39: PLOT 0,39 - Z
380 NEXT Z
382 PRINT : PRINT "YOUR SCORE=";
  GS,"TARGET SCORE=";TS
384 XL = CC:YL = CB
386 IF LE = 1 GOTO 182
388 GOTO 392
390 XN = 25:XR = 25:YN = 30:YR =
  30
392 GOTO 222
```

Main program loop.

```
502 L = L + CC: IF L / TT = CC THEN
  COLOR= CB: POKE 768,96: POKE
  769,56: CALL 770: POKE 768,1
  53: POKE 769,56: CALL 770: GOSUB
  162: GOTO 222
504 GOSUB 22
506 XN = XN + XM:YN = YN + YM
508 HW = SCRN( XN,YN): IF HW = C
  D OR HW = CH THEN 602
510 IF HW = CF THEN 302
512 R = R + CC: IF R > DI THEN R =
  CB
514 COLOR= CH: PLOT XN,YN: IZ(R) =
  OM
516 M = M + CC: IF M > DI THEN M =
  CB
518 IF IZ(M) = CF THEN XL = CB:Y
  L = CC: GOTO 526
520 IF IZ(M) = CE THEN XL = CB:Y
  L = CA: GOTO 526
```

```
522 IF IZ(M) = CD THEN XL = CA:Y
  L = CB: GOTO 526
524 IF IZ(M) = CC THEN XL = CC:Y
  L = CB
526 XR = XR + XL:YR = YR + YL
528 COLOR= CB: PLOT XR,YR
530 GOTO 502
```

Text page for a lost game. Calculate and display values.

```
602 TEXT : HOME : POKE - 16368,
  0
604 IF GS > HI THEN HI = GS
606 VTAB 3: PRINT TAB( 12)"SORR
  Y, YOU LOSE"
608 PRINT : IF HW = CD THEN PRINT
  TAB( 3)"YOU HIT THE BORDER
  OR AN OBSTACLE"
610 IF HW = CH THEN PRINT TAB(
  1)"YOU REVERSED DIRECTION OR
  HIT THE TAIL"
612 IF GC = 1 THEN GOSUB 802
614 IF GC / 5 = INT( GC / 5) THEN
  GOSUB 802
616 PRINT : PRINT TAB( 12)"YOUR
  SCORE IS ";GS
618 PRINT : PRINT TAB( 9)"THE H
  IGH SCORE IS ";HI
620 PRINT : PRINT TAB( 11)"I DR
  EW ";TP;" TARGETS"
622 PH = INT( (HT / TP) * 100)
624 PRINT : PRINT TAB( 7)"YOU H
  IT ";PH;"% OF THE TARGETS"
626 IF HT = 0 GOTO 632
628 TA%= GS / HT:TA$ = STR$( TA)
630 PRINT : PRINT TAB( 9 - INT
  ( LEN( TA$) ) / 2)"AVERAGE PO
  INTS/TARGET=";TA$
632 PRINT : PRINT "PUSH <RETURN>
  TO PLAY AGAIN, 'Q' TO QUIT"
  ;: GET A$
634 IF A$ = "Q" THEN 682
636 IF LE = CB THEN GOTO 646
638 IF LE = CC THEN GOTO 640
640 PRINT : GOSUB 92: PRINT "PRE
  SS ANY KEY TO DECREASE THE D
  IFFICULTY": PRINT TAB( 4)"P
  RESS ANY <RETURN> TO PLAY AG
  AIN": GET A$
642 IF A$ < > CHR$( 13) THEN L
  E = 0
644 GOSUB 72: GOTO 372
646 PRINT : GOSUB 92: PRINT "PRE
```

```

SS ANY KEY TO INCREASE THE D
IFFICULTY": PRINT TAB( 4)"P
RESS <RETURN> TO PLAY AGAIN"
;: GET A$
648 IF A$ < > CHR$( 13) THEN L
E = 1
650 GOSUB 72: GOTO 372

```

End of game.

```

682 HOME : VTAB 10: PRINT "THANK
YOU FOR PLAYING": PRINT : PRINT
"TRY AGAIN SOME OTHER TIME."
: END

```

Directions for play.

```

702 HOME : SPEED= 190
704 VTAB 8: GOSUB 92: PRINT TAB(
4)"- W H I T E L I G H T
N I N G -": PRINT
706 GOSUB 92
708 PRINT "DO YOU WANT DIRECTION
S (Y OR N)";: GET A$
710 IF A$ = "N" GOTO 750
712 HOME : GOSUB 42
714 PRINT " THE OBJECT OF 'WH
ITE LIGHTNING' IS TO SCO
RE AS MANY POINTS AS POSSIBL
E BY REACHING TARGETS THAT
WILL APPEAR RANDOMLY ON T
HE SCREEN.": PRINT
716 PRINT " YOU WILL DIRECT A
LINE ABOUT THE SCREEN.
EACH TIME THE LEADING END OF
THE LINE REACHES THE TARG
ET, YOUR SCORE WILL BE INCRE
ASED BY THE VALUE OF THE T
ARGET AND THE LENGTH OF THE
LINE WILL"
718 PRINT "BE INCREASED BY A COR
RESPONDING AMOUNT.": PRINT
720 VTAB 21: GOSUB 92: GOSUB 52
722 HOME : GOSUB 42
724 PRINT "ANY OF THE FOLLOWING
WILL CAUSE YOU TO LOSE THE
GAME:": PRINT

```

```

726 PRINT "(1) ALLOWING THE LEA
DING END OF THE LINE
TO CROSS THE BORDER OF THE
SCREEN.": PRINT

```

```

728 PRINT "(2) ALLOWING THE LEA
DING END OF THE LINE
TO CROSS THE 'TAIL' THAT WI
LL GROW AS THE GAME PRO
GRESSES.": PRINT

```

```

730 PRINT "(3) REVERSING DIRECT
ION BY 180 DEGREES. YOU
MUST REVERSE DIRECTION USING
TWO 90 DEGREE TURNS.
"

```

```

732 GOSUB 92: GOSUB 52

```

```

734 HOME : GOSUB 42

```

```

736 PRINT "USE THE FOLLOWING COM
MANDS TO MOVE THE LEADING E
ND OF THE LINE.": PRINT

```

```

738 PRINT "PRESS 'E' TO MOVE UP"
: PRINT : PRINT "PRESS 'D' T
O MOVE DOWN": PRINT : PRINT
"PRESS 'O' TO MOVE LEFT": PRINT
: PRINT "PRESS 'P' TO MOVE R
IGHT"

```

```

740 VTAB 21: GOSUB 92: GOSUB 52

```

```

742 HOME : GOSUB 42

```

```

744 PRINT " EXPERIENCED PLAYE
RS MAY WISH TO INCREASE
THE DIFFICULTY OF THE GAME.

```

```

THE SAME RULES APPLY WITH
ONE ADDITION.": PRINT

```

```

746 PRINT TAB( 4)"YOU MAY NOT A
LLow THE LEADING END OF": PRINT
TAB( 4)"THE LINE TO CROSS A
NY OF THE ADDED": PRINT TAB(
4)"OBSTACLES."

```

```

748 PRINT : GOSUB 92: GOSUB 52

```

```

750 PRINT : PRINT

```

```

752 GOSUB 92

```

```

754 PRINT "DO YOU WANT TO INCREA
SE THE DIFFICULTY?": PRINT :
PRINT TAB( 16)"(Y OR N)";:
GET A$

```

```

755 IF A$ = "N" THEN LE = 0: GOTO
758

```

```

756 IF A$ = "Y" THEN LE = 1: GOTO

```

```

758

```

```

757 PRINT : GOTO 754

```

```

758 HOME : VTAB 9: GOSUB 92

```

```

760 PRINT TAB( 5)"PRESS ANY KEY
TO PLAY THE GAME": PRINT : PRINT
TAB( 15)"GOOD LUCK!!"

```

```

762 PRINT : GOSUB 92

```

```

764 GOSUB 54

```

```

766 SPEED= 255: GOSUB 72: GOTO 3
72

```

POKEs for Machine Language sound routine.

```

782 POKE 770,173: POKE 771,48: POKE
772,192: POKE 773,136: POKE
774,208: POKE 775,05: POKE 7
76,206: POKE 777,01: POKE 77
8,03: POKE 779,240: POKE 780
,09

```

```

784 POKE 781,202: POKE 782,208: POKE
783,245: POKE 784,174: POKE
785,00: POKE 786,03: POKE 78
7,76: POKE 788,02: POKE 789,
03: POKE 790,96

```

```

786 GOTO 702

```

Music routine for lost game.

```

802 POKE 768,153: POKE 769,255: CALL
770: POKE 768,153: POKE 769,
255: CALL 770: POKE 768,153:
POKE 769,56: CALL 770: POKE
768,153: POKE 769,255: CALL
770: POKE 768,127: POKE 769,
255: CALL 770

```

```

804 POKE 768,137: POKE 769,56: CALL
770: POKE 768,137: POKE 769,
191: CALL 770: POKE 768,153:
POKE 769,56: CALL 770: POKE
768,153: POKE 769,191: CALL
770

```

```

806 POKE 768,162: POKE 769,56: CALL
770: POKE 768,153: POKE 769,
255: CALL 770

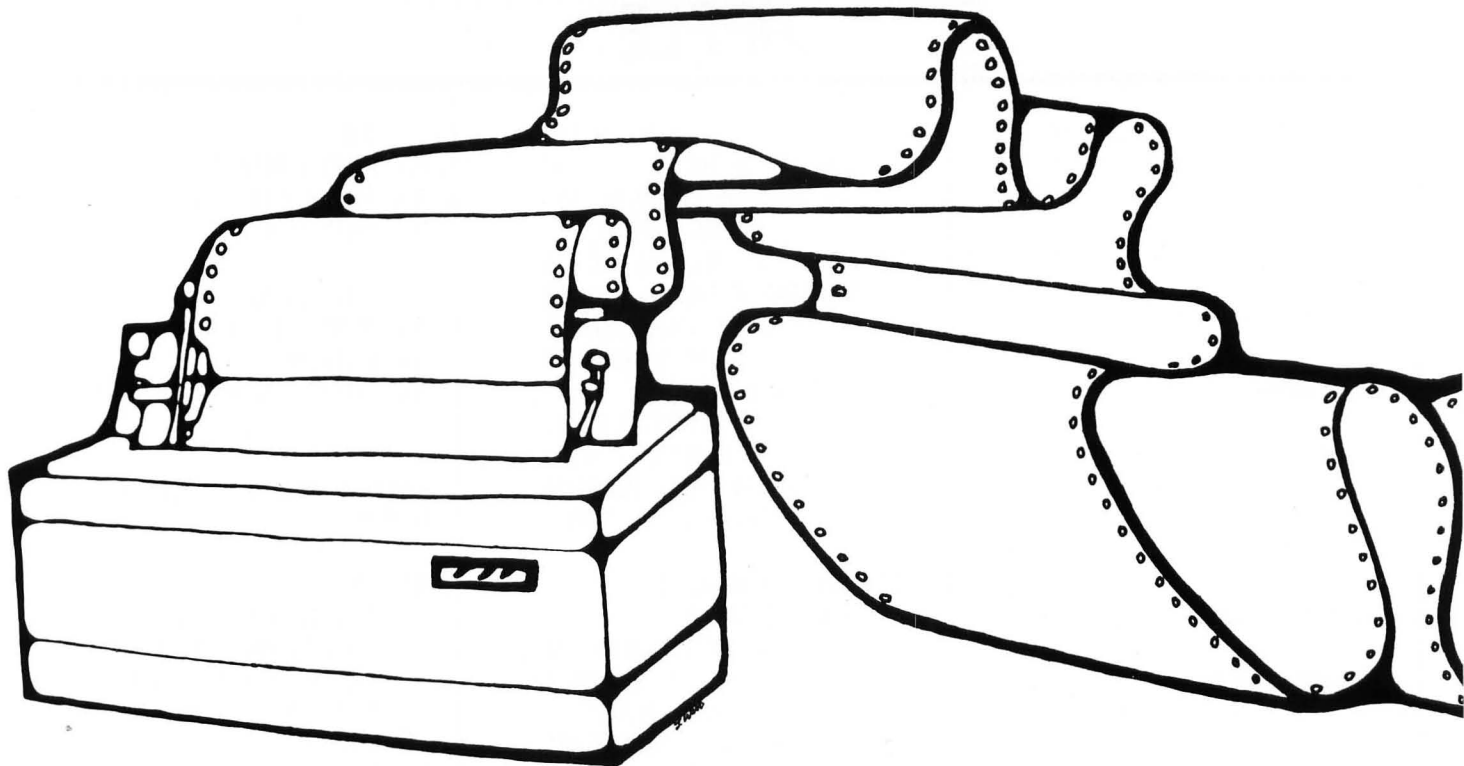
```

```

808 RETURN

```

5



APPLE™

by Fred J. Condo

Poster Maker is a large-character printing program for an Apple with Applesoft, 16K RAM, and a printer.

Programs that make banners have been around probably as long as there have been computers. But the banners have always run *along* the paper rather than *across* it. *Poster Maker* makes posters with words that run the same way ordinary printing would. It allows you to input a whole phrase, which it will print automatically, one word centered on each line. If you have an Epson MX-80 printer, you can choose the size of character you want by specifying the number of banner-sized characters per line. If any word won't fit, the program puts in a hyphen (but not necessarily in the grammatically correct position). If you just remember that the program starts a new line wherever you type a space, you can insert your own hyphen in a long word where it should be divided. To top it

all off, you can even specify whether each large character should be made up of the small characters that it represents, or of another character that you want to use.

The characters provided in the program are all the upper-case letters plus most of the punctuation available from the Apple keyboard. The character definitions are located in the DATA statements starting in line 10000. If you want to add or modify character definitions, here is the data organization: First, the character being defined. If in doubt, put this character in quotation marks. Next, 13 line definitions. There must be exactly 13, because the characters are made up of a 12 by 13 matrix. Each line definition is READ by the program (line 110) in number pairs, so each line definition must contain an even number of numeric values. Each of the line definitions must end with a pair of -1s, since this is the end-of-line flag. (A blank line is the pair of -1s alone.)

Each line definition consists of up

to three pairs of numbers (plus the terminal -1s). The first number of a pair is always a number of spaces to be printed; the second is a number of dots. Together these spaces and dots specify the printing pattern for a particular line of that character. The total number of dots plus spaces in each line must be no more than 12. The program itself does not check for faulty character definitions, so errors will show up either as SYNTAX ERRORS in the DATA lines or as scrambled lettering on the posters.

If you don't have an Epson printer, you will need to make a few small modifications to the program. If your printer can handle only one printing density (80 characters per line), then change line 210 to WIDE=6 and make the changes specified in the program documentation at lines 400-430. If your printer can handle alternate densities, then you'll need to make appropriate modifications of the character codes printed in those lines.

Poster Maker

APPLE™

Variables

A: Loop index in sorting routine.
A\$: Used by display subroutine to display centered lines on the monitor screen. Also used for Y/N replay at the end of printing.
B: Loop index in sorting routine.
C: The number of dots (characters) in a line definition.
CFLAG: If 1, user has specified a character to make up the large ones.
CH\$: Contains a character specified by the user to make up the larger characters. If null, characters are made up of their small analogs.
DESC%(x,y,z): Mnemonic for "DESCription;" contains the character definitions. X subscript is the character; y is the line number; and z is the item (number of spaces or dots).
DISPLAY: Used by display routine: 1 = normal, 2 = inverse, and 3 = flash.
HYFLAG: If 1, a word was hyphenated.

I: Item number during DATA READING; character number in P1\$ during printing; also a general loop index.
II: Item number during printing.
J, KK: Loop indices.
KK\$: Used in the input-simulation subroutine to accept each character as it is typed at the keyboard.
L: Saves LEN(L\$).
L\$: Contains each word of the input string as it is cut apart. (A word consists of the characters between spaces.)
L\$(x): Contains the characters defined in the DATA. X is analogous to the x subscript in the DESC% array.
LL\$(x): Same as L\$(x), but sorted into ASCII order. Used to present the available characters.
LTR: Number of characters defined, plus 1.
M\$: Used to save the results of MID\$ function calls.
OL\$: Mnemonic for "Old L\$." Contains the previous word printed, in order to determine if a hyphen was used. If so, centering

of the next line is suppressed.
P\$: String input by user to be printed.
P1\$: Same as P\$, but with undefined characters removed. This is the string which is cut up and put piece by piece into L\$.
QQ: Number of characters defined. Used to sort and display LL\$(x).
RS: String returned by the input-simulation routine.
S: Number of spaces in a line definition.
SS: Contains the number of spaces needed to center L\$ on a line.
TS: Temporary storage variable used to swap strings in array LL\$ during sorting.
VT: Used by display subroutine. Specifies VTAB number.
WIDE: Number of large characters per line.
WW: Used to get double-width characters from the MX-80. If the modifications to the program for non-MX-80 use have been made, this is always zero.
X: Loop index.

```

$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$  APPLESOFT BASIC  $
$  'POSTER MAKER'  $
$  AUTHOR: FRED J. CONDO  $
$  (C) 1982  SOFTSIDE  $
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

```

Set normal text mode and display title header.

```

10 NOTRACE : TEXT : HOME : SPEED=
   255: NORMAL
20 VT = 1:DISPLAY = 2:A$ = "
   POSTER MAKER      ": GOSUB
   750:VT = 3:DISPLAY = 1:A$ =
   "BY F.J. CONDO": GOSUB 750:V
   T = 4:A$ = "JULY 1981": GOSUB
   750
30 POKE 34,5

```

READ the character data into the arrays.

```

40 VT = 12:A$ = "READING DATA":DI
   SPLAY = 3: GOSUB 750
50 DIM DESC%(53,13,8),L$(53),LL$
   (53)
60 LTR = 1
70 READ L$(LTR):LL$(LTR) = L$(LTR)
   R): IF L$(LTR) = "XXX" GOTO
   160
80 VT = 13:DISPLAY = 1:A$ = "<" +
   L$(LTR) + ">": GOSUB 750
90 L = 1
100 I = 1
110 READ S,C
120 DESC%(LTR,L,I) = S:DESC%(LTR,
   L,I + 1) = C
130 IF S < > - 1 THEN I = I +
   2: GOTO 110
140 L = L + 1: IF L < = 13 GOTO
   100
150 LTR = LTR + 1: GOTO 70

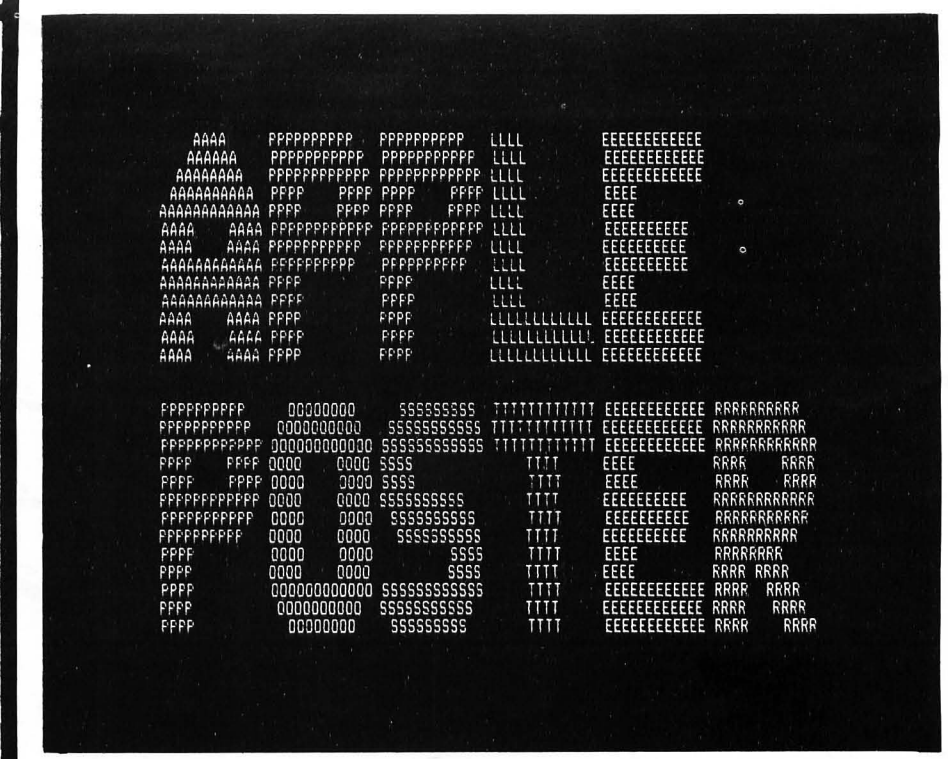
```

Sort the data in LL\$(B) in ascending order for later display of available characters.

```

160 HOME
170 VT = 12:DISPL = 3:A$ = "SORTI
   NG DATA": GOSUB 750
180 QQ = LTR - 1: FOR A = 1 TO QQ
   - 1: FOR B = 1 TO QQ - A: IF
   LL$(B) > LL$(B + 1) THEN T$ =
   LL$(B):LL$(B) = LL$(B + 1):L
   L$(B + 1) = T$

```



```
190 NEXT B,A
```

Input maximum line width. (For standard 80-column printers, use six characters per line.)

```

200 HOME : VTAB 12
210 INPUT "HOW MANY LARGE CHARAC
   TERS WIDE DO YOU WANT EACH
   LINE TO BE (MAX. 10)? ":WID
   E:WIDE = INT (WIDE): IF WID
   E < 1 OR WIDE > 10 GOTO 200
220 A$ = "": FOR I = 1 TO 21:A$ =
   A$ + " ": NEXT :VT = 2:DISPL
   AY = 1: GOSUB 750
230 VT = 2:DISPLAY = 2:A$ = "LINE
   S WILL BE " + STR$ (WIDE) +
   " LONG": GOSUB 750

```

Display available characters and input the words to be printed.

```

240 HOME : PRINT "USE ONLY": FOR
   A = 1 TO QQ: PRINT LL$(A):; NEXT
   A
250 POKE 34,8
260 HOME : VTAB 12: PRINT "PHRAS
   E ":; GOSUB 700:P$ = R$
270 IF P$ = "" THEN TEXT : HOME
   : END

```

Remove characters from the input string which are not defined in the DATA statements. P1\$ will then contain the phrase to be printed.

```

290 P1$ = ""
300 FOR I = 1 TO LEN (P$):M$ =
   MID$( P$,I,1)
310 FOR J = 1 TO LTR - 1
320 IF M$ < > L$(J) THEN NEXT
   J
330 IF J < LTR THEN P1$ = P1$ +
   M$
340 NEXT I
350 IF P1$ = "" THEN HOME : VTAB
   12: PRINT "NO DATA AVAILABLE
   FOR YOUR INPUT STRING PLEAS
   E TRY AGAIN, USING ONLY THE
   CHARAC-TERS LISTED ABOVE": FOR
   KK = 1 TO 1800: NEXT KK: GOTO
   200

```

Input the character to be used in forming the large poster characters.

```

360 CFLAG = 0: INPUT "CHARACTER T
   O PRINT OR <RETURN> TO MATCH
   PRINTING CHARACTER TO CHARA
   CTER BEING REPRESENTED: ":
   CH$:CH$ = LEFT$ (CH$,1)
370 IF CH$ < > "" THEN CFLAG =
   1
380 P1$ = P1$ + " "
390 HOME : VTAB 12: PRINT P1$

```

Lines 400-430 are specifically for the Epson MX-80. For a standard 80-column printer, delete 410-430

and delete everything in line 400 following "PR#1".

```
400 PR# 1: PRINT CHR$ (27); CHR$
(48);: REM COMPRESSED LINE
SPACING
410 IF WIDE > 6 THEN PRINT CHR$
(15): REM 132 CHARS/LINE
420 WW = 0: IF WIDE < 6 AND WIDE >
3 THEN PRINT CHR$ (15):WW =
14: REM 66 CHARS/LINE
430 IF WIDE < = 3 THEN WW = 14
```

Cut P1\$ into words (groups of characters separated by spaces).

```
440 PRINT
450 I = 1
460 L$ = ""
470 IF I > LEN (P1$) THEN PR#
0: HOME : VTAB 12: GOTO B20
480 M$ = MID$ (P1$,I,1)
490 IF M$ < > " " THEN L$ = L$ +
M$:I = I + 1: GOTO 470
500 I = I + 1
```

Center each word, unless the last word was hyphenated. (To center the trailing part of hyphenated words as well, delete line 520.)

```
510 L = LEN (L$):S$ = ""
520 IF RIGHT$ (OL$,1) = "-" THEN
HYFLAG = 1
530 IF L > WIDE THEN I = I - (L -
WIDE) - 2:L = WIDE:L$ = LEFT$
(L$,WIDE - 1) + "-"
540 IF L < WIDE - 1 AND NOT HYF
LAG THEN FOR X = 1 TO (WIDE
- L) / 2:S$ = S$ + " ": NEXT
X
550 OL$ = L$:HYFLAG = 0
560 L$ = S$ + L$
```

Print the current word and then go on to the next one.

```
570 FOR LINE = 1 TO 13
580 FOR Q = 1 TO LEN (L$):M$ =
MID$ (L$,Q,1)
590 FOR R = 1 TO LTR - 1
600 IF L$(R) < > M$ THEN NEXT
R
```

```
610 II = 1:SCOUNT = 0
620 S = DESCZ(R,LINE,II):C = DESC
%(R,LINE,II + 1)
630 IF NOT CFLAG THEN CH$ = L$(
R)
640 IF S < > - 1 THEN PRINT CHR$
(WW); SPC( S);:SCOUNT = SCOU
NT + S: FOR X = 1 TO C: PRINT
CH$;: NEXT X:SCOUNT = SCOUNT
+ C:II = II + 2: GOTO 620
650 IF SCOUNT < 13 THEN FOR X =
SCOUNT TO 12: PRINT " "; NEXT
X
660 NEXT Q: PRINT : NEXT LINE
670 PRINT : PRINT
680 GOTO 460
```

Subroutine to allow inputting of all characters, including commas, quotation marks, colons, etc.

```
700 R$ = "": PRINT "-->";
710 GET KK$: PRINT KK$;: IF KK$ =
CHR$ (13) THEN RETURN
720 IF KK$ = CHR$ (8) AND LEN
(R$) > 1 THEN R$ = LEFT$ (R
$, LEN (R$) - 1): GOTO 710
730 IF KK$ = CHR$ (8) AND LEN
(R$) < = 1 THEN R$ = "": GOTO
710
740 R$ = R$ + KK$: GOTO 710
```

Subroutine to display a string of characters centered on line VT. The value of DISPLAY determines if the string will be printed as normal, inverse, or flashing text.

```
750 ON DISPLAY GOTO 770,780,790
770 NORMAL : GOTO 800
780 INVERSE : GOTO 800
790 FLASH
800 VTAB VT: HTAB 20 - LEN (A$)
/ 2: PRINT A$: NORMAL : RETURN
```

Done with this poster; do another one?

```
820 INPUT "MORE? ";A$: IF LEFT$
(A$,1) = "N" THEN TEXT : HOME
: END
830 PR# 1: PRINT CHR$ (18) CHR$
(20): PR# 0: GOTO 200
```

DATA for characters. (See article for data format.)

```
10000 DATA A,4,4,-1,-1,3,6,-1,-1
,2,8,-1,-1,1,10,-1,-1,0,12,-
1,-1,0,4,4,4,-1,-1,0,4,4,4,-
1,-1,0,12,-1,-1,0,12,-1,-1,0
,12,-1,-1,0,4,4,4,-1,-1,0,4,
4,4,-1,-1,0,4,4,4,-1,-1
10001 DATA E,0,12,-1,-1,0,12,-1,
-1,0,12,-1,-1,0,4,-1,-1,0,4,
-1,-1,0,10,-1,-1,0,10,-1,-1,
0,10,-1,-1,0,4,-1,-1,0,4,-1,
-1,0,12,-1,-1,0,12,-1,-1,0,1
2,-1,-1
10002 DATA " ",-1,-1,-1,-1,-1,-1
,-1,-1,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1,-1,-1,-1,-1,-
1,-1
10003 DATA I,2,8,-1,-1,2,8,-1,-1
,2,8,-1,-1,4,4,-1,-1,4,4,-1,
-1,4,4,-1,-1,4,4,-1,-1,4,4,-
1,-1,4,4,-1,-1,4,4,-1,-1,2,8
,-1,-1,2,8,-1,-1,2,8,-1,-1
10004 DATA O,2,8,-1,-1,1,10,-1,-
1,0,12,-1,-1,0,4,4,4,-1,-1,0
,4,4,4,-1,-1,0,4,4,4,-1,-1,0
,4,4,4,-1,-1,0,4,4,4,-1,-1,0
,12,-1,-1,1,10,-1,-1,2,8,-1,
-1
10005 DATA U,0,4,4,4,-1,-1,0,4,4
,4,-1,-1,0,4,4,4,-1,-1,0,4,4
,4,-1,-1,0,4,4,4,-1,-1,0,4,4
,4,-1,-1,0,4,4,4,-1,-1,0,4,4
,4,-1,-1,0,12,-1,-1,1,10,-1,
-1,2,8,-1,-1
10006 DATA "-",-1,-1,-1,-1,-1,-1
,-1,-1,-1,-1,-1,-1,1,10,-1,-
1,1,10,-1,-1,-1,-1,-1,-1,-1,
-1,-1,-1,-1,-1
10007 DATA C,2,10,-1,-1,1,11,-1,
-1,0,12,-1,-1,0,4,-1,-1,0,4,
-1,-1,0,4,-1,-1,0,4,-1,-1,0,
4,-1,-1,0,4,-1,-1,0,4,-1,-1,
0,12,-1,-1,1,11,-1,-1,2,10,-
1,-1
10008 DATA R,0,10,-1,-1,0,11,-1,
-1,0,12,-1,-1,0,4,4,4,-1,-1,
0,4,4,4,-1,-1,0,12,-1,-1,0,1
1,-1,-1,0,10,-1,-1,0,8,-1,-1,
0,4,1,4,-1,-1,0,4,2,4,-1,-1
,0,4,3,4,-1,-1,0,4,4,4,-1,-1
10009 DATA S,2,9,-1,-1,1,11,-1,-
1,0,12,-1,-1,0,4,-1,-1,0,4,-
1,-1,0,10,-1,-1,1,10,-1,-1,2
```


Three Apple Text Editors

A comparative review by Jon Voskuil

Apple Writer II
from Apple Computer, Inc. Suggested retail price: \$150

SuperScribe II
from On-Line Systems. Suggested retail price: \$130

SuperText II
from MUSE. Suggested retail price: \$150

You may be considering buying one of the three text editor programs listed (in alphabetical order) above. They're not the only ones available for the Apple by any means, and the choice of these particular three should not reflect in any way upon the many others not reviewed.

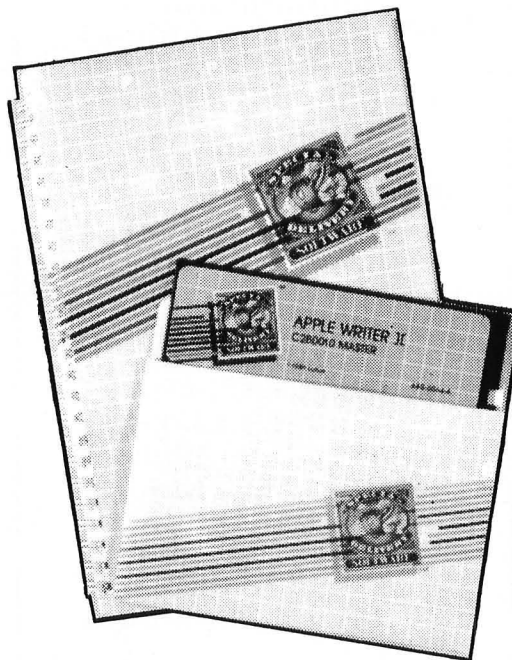
I feel it's only fair to tell you at the outset that I am much better acquainted with one of these software packages than with the other two. For over a year I have been working with *SuperText II* (because it happened to be the only word processor for the Apple that *SoftSide* had lying around when I came), and therefore know its capabilities pretty thoroughly. *Apple Writer II* and *SuperScribe II* have been in my possession only a few weeks. While I've been using them as much as possible to get a feel for them, there are many features that I simply haven't had the time to use. On the balancing side, though, it's the common, ordinary, everyday features of any given word processor that give it its "feel"; and the special features, which are not likely to be used as much, don't really affect that feel.

The common, ordinary, everyday business of a word processor is to make it easy to write and edit text. All three of these packages do an admirable job of that. They share a number of more-or-less standard writing and editing features which will woo you away from your electric typewriter forever. With any one of these you can:

Type any text into your computer's memory (in both upper and

lower case, even if your computer won't actually display lower-case letters), and transfer that text back and forth between the computer and a disk at will. All of them can take advantage of a simple modification which will allow you to use the shift key in the normal way to capitalize letters. If you have not made this no-cost modification, then capitalization is accomplished by pressing the ESC key before typing a letter to be capitalized.

Forget about carriage returns



when typing normal text, except at the end of paragraphs. If a word won't fit on a line, it will automatically be moved down to the next line as you type.

Correct typing mistakes as you type, simply by backspacing and retyping.

Move the writing cursor to any position in your text, and delete, insert, or change any characters you wish.

Move a block of text from one part of your document to another, without having to retype it.

Find each occurrence of any character or string of characters you're looking for in your text and, if desired, automatically replace those characters with other characters.

Print the text in memory on paper using a printer, in almost any conceivable format. Line spacing, justification, centering, margins, tabs, and page numbering are some of the parameters that can be specified.

The differences among these three word processors are to be found in two areas. They differ, first, in the approach they take to performing the basic editing functions. These differences are important, because they affect the way the program feels as you are using it. Are the keystrokes necessary to move the cursor around the screen and perform other editing functions "natural" ones? Or, at least, do they begin to feel natural after a bit of practice? This can be a rather subjective judgment, of course. Second, the programs differ in the features they add to the basic editing package. These might very well be the deciding factor for you in choosing one over the others. How much flexibility do you need? What kind of tradeoff are you willing to make between simplicity of use and versatility of application?

Apple Writer II

Apple Writer II comes with two identical 16-sector disks, a 106-page spiral-bound manual of the same dimensions as other Apple manuals, and a tear-out reference card with command summaries for editing and printing. The documentation is very well written and seems virtually error-free, typical of Apple

manuals in general. Someone who has never used a word processor before should have no trouble getting started with this friendly manual.

Apple Writer II is compatible with M & R Enterprise's Sup'R'Terminal 80-column board (and possibly others). Upon booting the program, this board will automatically be used if installed. Otherwise you are asked if you have a lower-case adapter installed. (If you don't, upper-case letters will be indicated by inverse characters, and lower-case by normal characters.) You are then shown a title page with a summary of frequently-used commands. Pressing RETURN clears the screen and sets you up to start typing in text. A "Data Line" at the top of the screen (which can be switched off if desired) keeps you informed about the number of characters in your text, the memory remaining, and other information.

If you are at all familiar with the built-in screen editing features of your Apple, you will feel somewhat at home from the start with the editing commands in *Apple Writer II*. Pressing the ESC key (twice, if you haven't done the shift-key modification) allows you to move the cursor in any direction through the existing text with the usual I, J, K, and M keys. Four similar keys under left-hand control (E, S, D, and X) move up or down 12 lines at a keystroke, or right or left a full word at a time.

One rather disconcerting feature of this mode is the manner in which the cursor moves through the text. *Apple Writer's* cursor never actually occupies the same location as a character in the text. When you're typing normally, of course, it's always just to the right of the characters you're entering. But when you move it through text already entered, it squeezes itself between two existing characters. Moving left and right across a line, then, gives an interesting ripple effect. The ripple effect when moving up and down from line to line, however, takes a little more getting-used-to. If you move the cursor to a line which is already full, the last word of that line is bumped down to the next — which may in turn cause a word to be bumped from that line

to the next, and so on. This makes for quite a bit of visual chaos on the screen, making it difficult to keep track of where you are in the text when moving rapidly from one line to the next.

The left and right arrows also function as editing keys. The left arrow deletes text backward, and the right arrow re-enters deleted characters, much as they do when editing BASIC program lines. The difference here is that the left arrow actually erases characters from the screen when deleting them, and the right arrow rewrites them. This provides a simple method for moving a portion of text from one place to another: You position the cursor at the end of the text you want to move, delete it from that place using the left arrow, move the cursor using the appropriate ESC key sequence, and then "regurgitate" the deleted text at the new location using the right arrow. This approach is a bit unconventional, but quite ingenious.

All other word processing functions are accessed through control characters. Insertion and deletion of text, for example, can be done a full word or a full paragraph at a time instead of just one character at a time. CTRL-W deletes or inserts a word, while CTRL-X does the same for a paragraph. Whether you're inserting or deleting is controlled by the direction of a little arrow on the Data Line, which is toggled right or left using CTRL-D. You can move directly to the beginning or end of your text by pressing CTRL-B or CTRL-E. Pressing CTRL-R switches you into the replace mode, allowing you to type over existing text with new. CTRL-I moves you to the next tab position, which you set earlier using CTRL-T. Finding and replacing certain characters in your text is accomplished by pressing CTRL-F, followed by the characters you want to find and (optionally) the new ones you want to put in their place. Other control characters allow you to save and load files from disk, clear the current text from memory, convert upper-case text to lower-case and vice-versa, execute normal DOS commands, and access "help" displays which explain various commands.

Two powerful editing features of *Apple Writer II* bear mentioning. One allows you to define a "glossary" of words, phrases, or any other characters, which you may have occasion to use frequently while writing. Once defined, pressing three keys (CTRL-G plus one other which you specify in the definition) will insert the entire defined term into the text. A single term can be up to 128 characters, and the whole glossary table can be up to 2048 characters long. Any number of glossary tables can be created and stored on disk.

A second powerful feature allows you to split the screen display into upper and lower sections, and display different parts of your text independently in the two areas. This can be very helpful when you need to refer one portion of text while writing or editing another. (Along these same lines, another option allows you to view the contents of a file on disk while not disturbing the file in memory.)

In addition to being a capable editor, *Apple Writer II* is also a capable printout formatter. Printout formatting can be done either from within the text by use of embedded format commands, or using a print options menu prior to printout time. This menu's default values can be reset at any time, and permanently saved on disk. All the standard options are available, including left, right, and center justification options, paragraph indenting (or "outdenting"), pause between pages for sheet-feeding your printer, margin values, and page length, and automatic page numbering. Also available are automatic top and bottom line headings.

A major feature of *Apple Writer II* not yet mentioned is WPL. That stands for "Word Processing Language," and it represents a method of automating various word processing tasks. If you are familiar with the use of the Apple DOS EXEC command, then you understand the basic idea of WPL. WPL allows you to create a disk file containing commands which automatically control the operation of the word processor. Once you create this command file, you can use *Apple Writer's* DO command to

tell the computer to take over, using that set of commands.

The most common application of such automation would be in the creation of form letters. You would first create your standard form letter. Instead of typing a specific name, address, date, and so forth in the letter, you would enter generic phrases of your choice at the correct places. You would then create (or convert to a format usable by *Apple Writer*) a text file containing your names and addresses and whatever other information you want. Finally you would create a WPL program which would automatically go through the name/address file entry by entry, inserting information where indicated in the form letter, and print out each letter.

Various conditional operations can be included in your WPL program. This would enable you, for example, to insert a certain paragraph in the letter if a person's zip code were within a given range. Keyboard input can also be programmed into the WPL program, to give you the capability of making run-time choices about the exact form and content of the letters.

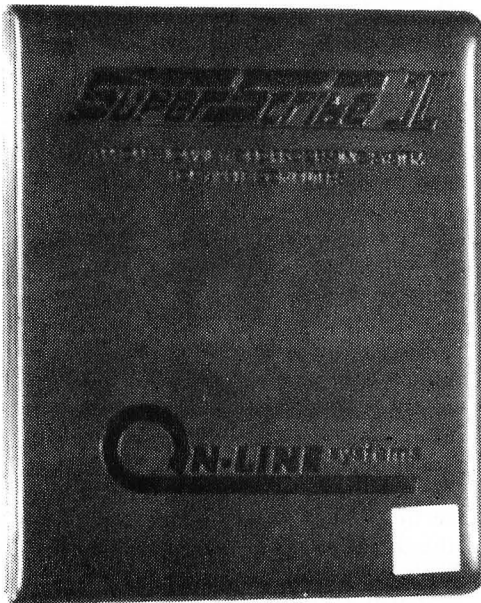
If WPL sounds versatile, it is. If it sounds a little complex, it is that too. A LITTLE complex. It's the sort of thing that programmers will rather enjoy, and non-programmers might find intimidating. It requires learning a very simple programming language consisting of just 17 commands, and then putting those commands together (very much as you would in a BASIC program) to control the operation of the word processor. You can even use some WPL capabilities without learning the commands, since there are five WPL programs included on the *Apple Writer II* disk (including one for printing form letters).

Apple Writer II stores both text and WPL programs in normal text files on normally formatted 16-sector disks. This makes it easy to back up important files or whole disks, using a copy program such as Apple's *FID*. (The master and backup disks supplied are in 16-sector format, but are protected against normal copying methods.) A convenient feature allows you to INITIALIZE a blank disk directly from the editor, in case you run out

of disk space in the middle of an important document. This procedure produces a data disk which will not boot by itself but which gives you 8K more available space than a normally initialized disk. Other DOS commands — CATALOG, RENAME, LOCK, UNLOCK, and DELETE — can also be executed from the editor without having to exit the program.

SuperScribe II

A step up the alphabet from *Apple Writer II* is *SuperScribe II*. Like *Apple Writer*, it comes with two 16-sector disks, a manual, and a refer-



ence card. In this case the manual is a little thicker (154 pages) and has a ritzier cover (a padded loose-leaf binder), and the reference card is a folded affair with three sides of information rather than two. The documentation is pretty well written, but rough in places — not quite as ritzy as its cover. The roughness is minor but occasionally annoying, consisting mostly of improper punctuation which gives some sentences a different meaning than was intended. Overall the writing style is not quite as lucid as *Apple Writer's*.

To be fair, it must be pointed out that describing *SuperScribe II* is a more complex task than describing *Apple Writer II*. Its features are more numerous and flexible, and therefore require more extensive knowledge to use effectively. A

quotation from the manual itself highlights the tradeoff that one makes for increased flexibility: "As you have seen, *SuperScribe II* offers you a great deal of flexibility. It can be used to handle the vast majority of your writing needs. The problem with any program which has such flexibility is that there is a tendency for it to be complicated to use for simple tasks." (p. 73) It goes on to describe a method for reducing that complexity, by creating a number of standard formats which you commonly use and saving these on the master disk. This is a good suggestion, and one that will simplify the use of any word processor. The fact remains that a larger choice of options tends to make overall operation more complicated.

SuperScribe II is not designed to make use of 80-column boards or of lower-case adaptors. Such hardware modifications are superfluous, because it generates its own character set using software. That set includes all standard alphanumeric characters and punctuation, plus special symbols representing control characters and carriage return. These are all quite legible and attractive, complete with descenders on the lower-case letters. This is a very (very) nice feature, since working with an upper-case-only display is aggravating, and lower-case adaptors aren't free.

What makes it even nicer is that, in addition to the normal 40-column character display, you can switch to a narrower character set which squeezes 70 columns onto one line. If you have a good black and white monitor, you can make use of this 70-column display as your normal mode. With a color monitor or a normal TV set, the narrower characters are only marginally readable and would be very tiring for continuous use. They are still useful, however, for previewing the way the text will look on a normal printed page, or for constructing charts and tables where you need to visualize everything in vertical columns all the way across the page.

When you boot the *SuperScribe II* disk, you will be confronted with the main menu, giving you the choice of entering the Editor or the Runoff program. Presumably the two parts of the word processor are



separated in this way because of memory requirements. (After all, an 8K chunk is taken out of text memory because of using the Hi-Res screen to display the software-generated characters.) It is in some ways a regrettable separation, because it makes the process of getting a printout much more cumbersome.

With most word processors, a few keystrokes and a few seconds will take you from editing text to printing it out on a printer. This one requires that you save the text to disk, exit the editor, and enter the runoff program to obtain your document in final form. In addition to the 30 seconds required to load Runoff, you must then re-enter the name of the disk file that you want to print before you can actually get to the printing. Then, if you find that you want to make the most minor of modifications to the text after you've seen the printout, you must rerun the Editor (30 seconds), re-enter the disk file name and load the file, make the changes, save the text again, rerun Runoff, and again re-enter the disk file name. (Incidentally, a minor gripe is that after any disk operation, the cursor always returns to the beginning of the file instead of to the location where you were working.)

Some relief from this cumbersome process is found in a feature that allows you to do a rough-form printout directly from the editor. This listing feature allows you to specify only two format parameters: line length and spacing. So, while it is certainly helpful in getting an idea of how the text will look, it's not a substitute for the final product.

The editor itself is a very capable piece of software, with enough features (I would think) to satisfy almost any writer. All the capabilities mentioned above for *Apple Writer II* are also available with *SuperScribe II*. The editing functions, however, do not have as familiar a feel at first. Instead of using ESC key sequences to move the cursor about, CTRL and SHIFT-CTRL combinations are used, along with the left and right arrows.

The arrow keys do not delete and rewrite text, but merely move the cursor left and right over the characters without effecting them.

To move the cursor vertically, the key combinations CTRL-A and SHIFT-CTRL-A are used. Following a convention used throughout *SuperScribe II*, the unshifted combination moves forward in the text (down, in this case) and the shifted combination moves backward (up). In like manner, CTRL-Z and SHIFT-CTRL-Z move the cursor a word at a time, and CTRL-P and SHIFT-CTRL-P move it one full screen page at a time. Deleting text is accomplished (in a more conventional manner than with *Apple Writer*) character by character with CTRL-D or SHIFT-CTRL-D, and line by line with CTRL-K or SHIFT-CTRL-K.

This approach to editing is rather satisfying logically, but does have its problems. The chief problem is that it sometimes requires relatively complex keystroke patterns. To move the cursor up a line in the text (a very common operation), you must hold down three keys simultaneously: SHIFT, CTRL, and A. To move up repetitively, you need to hold down a fourth, the REPT key. I have yet to find a thoroughly comfortable way of doing this, although I'm getting somewhat used to it. If you haven't made the shift-key modification to your Apple, using this editor will impel you to do so. (I did so after half an hour's practice without it. The manual contains how-to instructions, but it's a bit unclear which of the solder points on the underside of the shift-key switch to use. Naturally, I picked the wrong one the first time — with no ill effects, except for all my text being upper case.)

One mark of *SuperScribe's* versatility, however, is its ability to accept almost any degree of personal customizing. If you think you can work out a better system of moving the cursor around the screen using different combinations of characters, you're free to try. In a way similar to *Apple Writer's* glossary setup, you can redefine any key or key combination to represent any other key or key combination. So if you'd rather move the cursor up a line using CTRL-S instead of SHIFT-CTRL-A, you just punch a few keys and it's done. Of course, in doing so you lose the ability to move the cursor to the next occurrence of

a given character in a line, which is what CTRL-S normally does. If you want to retain that function, you have to redefine it first in terms of another key combination. Judicious use of such "macros" can be of great benefit, both in redefining editing functions and in quickly entering often-typed words and phrases.

SuperScribe II has several additional capabilities which should be of considerable interest to serious writers. One is its ability to work with text files larger than the available memory inside the Apple, without the writer having to worry about managing several disk files. It does this by automatically reading from and writing to the disk as needed. It's a little disconcerting the first time this happens: The disk drive starts whirring and the keyboard freezes for a second or two, and you're sure that you've been zapped by your friendly neighborhood arc welder. But then the cursor returns, all the characters you typed during that second or two appear on the screen in the right place, and all is well. You can thus work with a file as large as an entire disk's storage capacity.

Another plus for serious writers is the ability to maintain up to four separate indices of key words or subjects in your document. As you write, you can insert an index token character next to any word or phrase that you want to be indexed, and when you actually print out the text all such references will be noted by page number. You could also, if you care to, automatically index every occurrence of a given string of characters using the find/replace function to insert the appropriate index token. The whole system is a flexible one, allowing you to keep track not only of words actually printed in the text, but also of themes or ideas which you may want to index.

Other features include automatic printing of multiple page headers and footers, automatic or manual hyphenation, and the ability to accept footnotes and print them on the proper page of the finished text. You also have the ability to "spool" a text file from a disk to a printer, while working on a different text file with the editor. This can be a signifi-

cant time-saver if you need to print out a very long file.

Complete form-letter capabilities are also built into *SuperScribe II*. Unlike *Apple Writer*, you do not need to use a separate control program to execute the necessary repetitive functions; they are all built into the Runoff program. Using a text file containing names and addresses, and another containing the form letter itself, you can automatically print "personal" letters to your heart's content.

Like *Apple Writer II*, *SuperScribe II* uses standard 16-sector formatted disks for text storage. The master and backup disks are not copiable by normal means, however. (You can copy all the files, but you can't create a bootable disk.) Unlike *Apple Writer*, you cannot initialize a blank disk from the editor, a minor inconvenience. Incidentally, the *SuperScribe* disks are very sensitive to disk drive misadjustments, due to the complex nature of the disks' architecture (their copy-protection scheme). One of my drives, which handles other commercial disks almost flawlessly (I admit there is an occasional boot error), will successfully load the editor and runoff programs only about a third of the time.

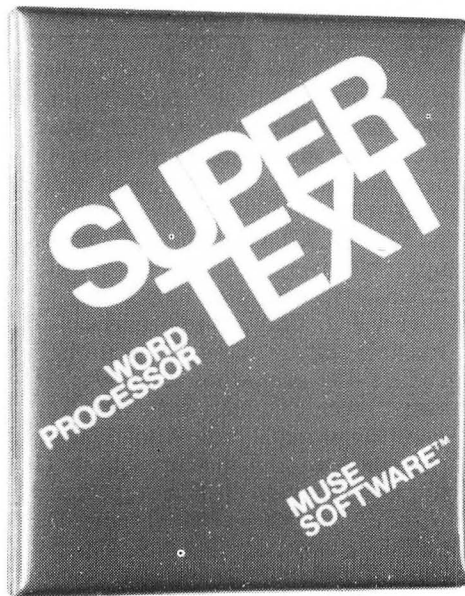
SuperText II

Finally we come to *SuperText II*, a bit beyond *SuperScribe II* in both the alphabet and the price scale (but not by much, in either area). It comes with two identical 13-sector disks which will boot also on 16-sector systems, a 90-page manual in a moderately ritzy padded binder (same page size as the others), and no rip-out reference card. The quality of the documentation is excellent, on par with that of *Apple Writer II*.

This editor's functions are divided clearly into three writing/editing modes, a unique math mode, and a printout mode. The writing/editing modes, in which you spend the vast majority of the time, are the cursor mode, the add mode, and the change mode.

The cursor mode is the one in which you find yourself upon entering the editor, and from which all other modes and operations are accessed. From this mode, moving the cursor around in your text is very

convenient. Like *Apple Writer's* I-J-K-M diamond of cursor movement keys, *SuperText II* has its own diamond: The left and right arrows, RETURN, and /. Being close to the REPT key, these are very convenient to use and quickly become second-nature. For grosser movement through the text, there are logical extensions: Pressing the ESC key prior to any of the four basic movement keys causes the cursor to move all the way to the left, right, top, or bottom margin of the screen display. Beyond this, pressing P moves you one screen page forward or backward through the text; the



direction is determined by a direction indicator which is changed using the + and - keys (shifted or unshifted). You can also shift the text display just one line up or down by pressing the L key, move the cursor to the center of the screen or to the location of the last change in the text with one keystroke, or move all the way to the beginning or end of the file by pressing ESC-- or ESC-+.

It is from the cursor mode also that you do such editing operations as deleting (although you can also delete from the change and add modes), finding and replacing strings of characters, and moving blocks of text from one place to another. For instance, CTRL-D deletes a single character, ESC-

CTRL-D deletes to the end of a word, and CTRL-G deletes to the end of a line. CTRL-F is used to find a string, and CTRL-R to replace one string with another. Blocks of text are manipulated in a manner similar to *SuperScribe II* and other word processors: A block marker is inserted at the beginning and end of the section of text, and then it can be copied to another place, deleted, or saved to disk as a unit. Such block markers, as well as all other embedded control characters, are designated by inverse video letters on the screen (in contrast to *Apple Writer*, which uses inverse to designate capital letters and flashing video to indicate control characters).

Like *Apple Writer*, *SuperText II* has a split-screen mode which allows viewing and manipulating two different sections of text simultaneously. A convenient additional feature allows the split in the screen to be moved upward or downward, to be able to view more of one section than the other. This function, too, is manipulated from the cursor mode.

From the cursor mode you can enter any of the other modes or execute disk operations, each through a different control code. CTRL-L and CTRL-S enable you to load, save, and otherwise manipulate disk files; CTRL-A enters the add mode; CTRL-C enters the change mode; CTRL-E enters the math mode; and X enters the printout mode.

The add mode is the basic writing mode, and operates in much the same way that *Apple Writer's* basic writing mode does. Anything you type is added into the text wherever the cursor is, pushing any text ahead of it forward as you go. The left arrow deletes text already typed, but doesn't save it for re-entry elsewhere. The right arrow moves to the next location on the current line which is directly under a space on the line above — an "implied tab stop." ESC-SPACE, ESC-/, and ESC-; are variously used to fill a line with spaces or to copy one or more words from the previous line into the current line. Several other tabbing and word-justification commands are also available. The only macro capability is the optional use of the colon key to enter "the" into the text. Pressing ESC twice returns

you to the cursor mode.

The change mode is very similar to the cursor mode, except that characters at the cursor can be changed by typing over them. The only characters which can't be substituted in this mode are /, carriage return, CTRL-D, and CTRL-G, since they are interpreted as cursor movement and editing commands. Return to the cursor mode is again via ESC-ESC.

The printout mode provides the usual opportunities to format the final printout of your file, either through use of the default options which can be set upon booting the disk, or with embedded format lines and other control characters to modify those default values. Six control characters are user-definable, to represent any control sequence that your printer may require for special printing functions. This doesn't represent as much versatility as the other editors (and setting up the definitions involves a bit of hexadecimal arithmetic and use of the Apple Monitor to modify memory) but it seems adequate for most users.

A preview feature in the printout mode is a convenient option that allows you to view the text on the screen exactly as it will be printed on paper, with line spacing, page breaks, page numbers, and justification. Of course, you can't view 60 columns on a 40-column display, so lines wider than 40 characters wrap around to the beginning of the next screen line. But the preview does give an accurate impression of the final form of the printout, before wasting any paper.

SuperText II allows you, both in printout and in editing modes, to link any number of disk files together so that they are treated essentially as one file. This method is not quite as transparent to the user as *SuperScribe's*, but it works effectively. Multiple parts of a large document, then, can be printed sequentially, and edited in either forward or reverse sequence.

A unique feature of *SuperText II* is its math mode. It allows you, while in the process of preparing a document with the editor, to do various mathematical calculations. With 15-digit capacity, it is capable of almost any calculations you would normally encounter. It can be

used either directly as a calculator, by simply entering numbers and functions and looking at the displayed result; or, more significantly, to perform calculations on numbers in your text file.

If you have a table or chart, for instance, with columns of figures, and need totals for each column, it's a simple matter to position the cursor over the bottom number, press C or CTRL-C, and see the total appear in the accumulator. This can then be inserted at the proper place in the text by moving the cursor and pressing R. Numbers not in columns can also be used in calculations. Just move the cursor to a number, press a mathematical function key, and that function will be executed using the accumulator total and the number at the cursor. This mode would be extremely helpful for someone who prepares financial, engineering, or other sorts of reports using a word processor.

SuperText II has one distinctive feature which may be either a matter of indifference or a matter of some importance to you. It does not use standard disk formatting. Disks which are to be used to store text must be formatted using the *SuperText* disk. It also stores text on disk not in a text file format but in a binary file format. The significance of this is that its disks and files are essentially incompatible with everyone else's disks and files. This is of very little or no importance if it is the only word processor you use, and if you have no occasion to want to edit or merge other types of files into your normal word processor files. (The *SuperScribe II* manual gives instructions for an indirect method of loading *SuperText* files into the *SuperScribe* editor, assuming that you have both word processors.)

If, on the other hand, you want to be able to load and edit, say, Applesoft programs (saved in non-tokenized form as text files), you just can't do it with *SuperText*. And that goes for any other disk files which are saved as text files on normally formatted disks. I don't find a word processor to be a very convenient program editor. (The combination of Synergistic Software's *Program Line Editor* and Highlands Computer Services' *Co-Resident*

Applesoft Editor is unbeatable in this department.) But if you want to do ANY fooling around with normally formatted text files, be forewarned.

Conclusions and Recommendations

Here are my gut-level reactions to these three products, all of which are excellent and apparently bug-free.


SuperText II is a comfortable and faithful old friend which I have found easy and sensible to use. It doesn't have the kinds of frills that *SuperScribe II* does, but it has enough to make it perfectly acceptable for everyday word processing and the occasional more lengthy and demanding work I've done.

Apple Writer II is an elegantly simple editor, with which an Apple programmer can feel at home almost instantly. Its Word Processing Language gives it automated capabilities which raise its basic simplicity to a much greater level of sophistication for those who need it.

SuperScribe II is at once the most exciting and the most aggravating of the three. I like all its ingenious features, but dislike some aspects of its basic personality. I definitely have a love/hate relationship with this program.

I can recommend all three as excellent performers. My guess is that serious writers might be happiest in the long run with *SuperScribe II*, although they might find its complexity a hindrance at times. The more casual writers might be happier with *Apple Writer II* or *SuperText II*. Of these, *SuperText* uses what I perceive as a more conventional approach to editing, although either would be easy enough to adopt in short order.

Form letter capability is not included with *SuperText II*; however, a form letter/address book module is available (and how many of us need to generate form letters anyway?).

There are undoubtedly other packages out there that offer equally reliable basic editing, and may have special combinations of features that you need. But I don't think you can go too far wrong with one of these. 

K-Byter

Crypto

An ATARI® K-Byter by Jerry Aamodt, Rochester, MI

When you RUN this program the computer will select a phrase from its repertoire, encode it, and display the encoded phrase twice on the screen. To decode it you enter the letter to be changed and, when prompted, the letter to be substituted. The first line displayed on the screen will be changed according to your command; the second line will remain in its original form for reference. When the decoding is completed correctly, the computer will select another phrase and continue.

A branch between lines 40 and 50 could be used to allow the input of a phrase (A\$) up to the DIMENSION you wish to set. You could also allow a coded phrase to be entered (B\$), and then use the program to decode it — in which case the IF...THEN... in line 130 should be deleted or circumvented.

Of course, the DATA statements may be freely changed and supplemented. When changes are made, just make sure that the random number multiplier in line 40 (5, in the listed program) is changed to equal the total number of phrases in the DATA lines.

```

5 REM CRYPTO by Jerry Aamodt
10 DIM A$(300),B$(300),C$(300),D$(1),E
$(1),A(26)
20 FOR I=1 TO 26:A(I)=-1:NEXT I:C=65:B
$="":FOR I=1 TO 26
30 B=INT(26*RND(0)+1):IF A(B)<>-1 THEN
30
40 A(B)=C:C=C+1:NEXT I:RESTORE 1000+IN
T(5*RND(0)):READ A$:B#=A$
50 FOR I=1 TO LEN(A$):IF ASC(A$(I,I))>
64 AND ASC(A$(I,I))<91 THEN B$(I,I)=CH
R$(ASC(A$(I,I))-64)
60 NEXT I:?"":C#=B$:POSITION 2,6:? C
$:POSITION 2,15:? B$
100 POSITION 2,20:? "ENTER LETTER TO B
E CHANGED";:INPUT D$
110 POSITION 2,20:? "ENTER LETTER TO B
E TRIED ";:INPUT E$
120 FOR I=1 TO LEN(A$):IF B$(I,I)=D$ T
HEN C$(I,I)=E$
130 NEXT I:POSITION 2,6:? C$:IF C#=A$
THEN 20
140 GOTO 100
1000 DATA NOW IS THE TIME FOR ALL GOOD
MEN TO COME TO THE AID OF THEIR COUNT
RY
1001 DATA YOU'VE COME A LONG WAY BABY
1002 DATA SING A SONG OF SIXPENCE POCK
ET FULL OF RYE
1003 DATA I'VE NOT YET BEGUN TO FIGHT
1004 DATA I'VE GOT THE WORLD ON A STRI
NG

```


Renumbering

For The ATARI®

by Frank Roberts

Renumbering for the ATARI® is a utility program for a 16K ATARI® 800 with disk drive. It is included as a bonus program on this month's ATARI® DV.

Have you ever allowed "plenty" of room between lines only to find later all the space had been used in earlier revisions? What does a programmer do when an insertion is needed between lines 1873 and 1874? Renumbers the program, right? After all, BASIC doesn't allow fractional increments and, alas, ATARI® BASIC has no renumbering command.

At times like these (and I have encountered many), renumbering a program by hand is painfully necessary. The commercial ones I have seen on the market start at around \$20. Well, relax. Here's a cheap and easy renumbering program for the ATARI®.

The secret to it is in the LIST"D:" command. A program "SAVED" to the ATARI® disk is written in a tokenized version (speedy and efficient for the com-

puter, incomprehensible to most humans), but a LISTed program is written to the disk exactly as it is entered: a BASIC listing containing all line numbers, variables and word commands.

By using the GET command, each element of the program can be examined in order of its listing. Before RUNning this program *the program to be renumbered must be LISTed to the Disk*, then the renumbering program LOADED into RAM and RUN. From there on it is all automatic.

The program is in two parts. Part one calls the target program from the disk and places the old numbers into a pseudo string array, then calculates the new line numbers in accordance with user instructions and places them into a corresponding string array (strings take a lot less RAM than numerical arrays).

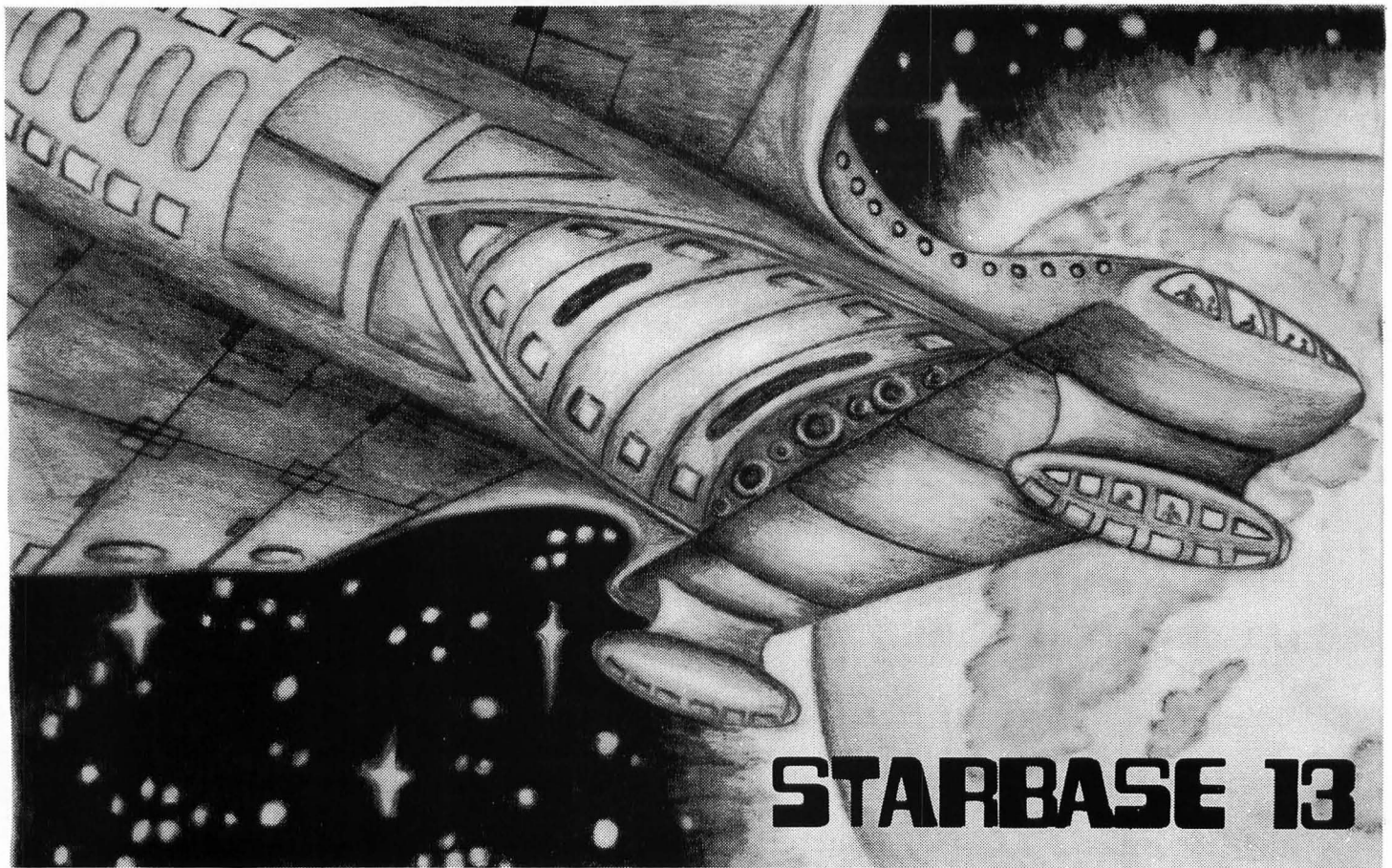
The second part searches each line for GOTO, GOSUB, THEN and TRAP, and for the line number immediately following each, and matches it with the old destination number. The new number is then

located in the NU\$ and placed in the original line. The edited line is then filed in a temporary file (D:T).

When the program has been renumbered, the XIO commands purge the old listing and rename the newly-numbered version, at the user's request. Once this is done, the newly-renumbered program should be ENTERed and the SAVED to the disk before RUNning.

The program is slow, but effective; it is inexpensive and has proven itself indispensable to me. On long programs I just take a break with a cup of coffee while the ATARI® is doing my "dirty work."

Numeric variables and calculated commands such as GOTO X or GOSUB X*1000 will need to be reset manually. The program allows easy access to such lines by recording the lines containing such reroute commands in string array (M\$). At the end of renumbering, these line are printed for the convenience of the programmer. The program also allows for optional printer listing of the renumbered version. ☺



STARBASE 13

ATARI®

by Mark Lewis Baldwin

Starbase 13 is an space game for the ATARI® 400/800 computer with a minmum of 16K RAM.

You are the commander of Starbase 13, the last defensive outpost between Earth and the dreaded alien invasion fleet. The longer you hold out, the more time Earth has to prepare its defenses, so you must stay and fight to the last humanoid.

In order to defend yourself, you have a giant laser cannon capable of penetrating any craft's force shield with a single shot. The joystick is used to change its direction while the trigger button activates firing. But beware, you only have a limited amount of energy. As you fire the cannon your shields weaken and the enemy gets closer.

The aliens have been well trained in attack and flee maneuvers. They will appear at the outer limits of your shields, fire their weapons and retreat. As they gain experience in attacking you, they remain visible for shorter periods of time, so be quick.

To start the game, push the trigger on your joystick controller and off you'll go. Your score is derived from the number of alien ships and missiles destroyed, plus the time you remain alive.

For a change of pace, insert another joystick controller in slot 2 and give it to a friend. Then push any console button and you will hear two beeps denoting the two-player mode. Push it again to return to the one-player mode (one beep). While in the two-player mode, player one controls the laser while player two fires. This can be rather tricky.

The following are a few comments about the program coding: The use of almost duplicate sets of coding several places in the program is not a result of poor programming. This was done to gain as much speed as possible out of a program with a very slow language. Also the explosion subroutine at 5300 might be of interest. It allows a long explosion to cycle while the rest of

the program can still continue to execute.

Variables

AF: Type laser hit.
 ALNP1: Position of alien 1.
 ALNP2: Position of alien 2.
 ALNT1: Time to event, alien 1.
 ALNT2: Time to event, alien 2.
 ARNG: Current shield range.
 AT: Alien reaction time.
 BASE: Position of base laser cannon.
 CONSOL: Address of START buttons.
 E: Energy counter.
 ESTP: Energy shield loss rate.
 FF: Laser fired flag.
 HF: Hit flag.
 MI(n): Status of missile n.
 MS: Missile speed.
 P: Stick request.
 R: Laser range.
 S3: Explosion status.
 SC: Score.
 T: Time counter.
 TSPD: Rate of change of AT.

ATARI®

```

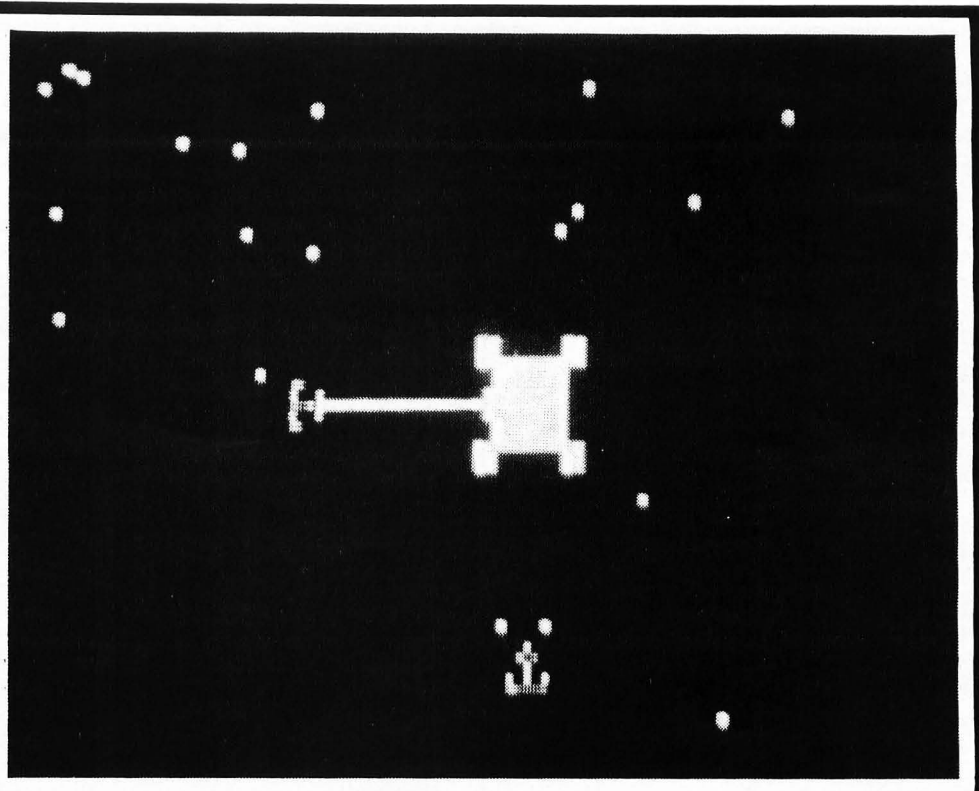
#####
$ Atari BASIC $
$ 'STARBASE 13' $
$ AUTHOR: Mark Lewis Baldwin $
$ (c) 1982 SofttSide $
#####

```

```

5 DIM MI(4):CONSOL=53279
10 GRAPHICS 0:SETCOLOR 2,6,1:SETCOLOR
4,6,1:UB=PEEK(560)+PEEK(561)*256+4:POK
E UB-1,70:POKE UB+2,7:POKE UB+3,7
20 FOR I=4 TO 8:POKE UB+I,6:NEXT I:POK
E UB+22,65:POKE UB+23,PEEK(560):POKE U
B+24,PEEK(561):SETCOLOR 3,8,6
30 POSITION 4,1:? #6;"STARBASE 13"
40 POKE 87,1:POSITION 2,7:? #6;"the la
st stand!"
50 POKE 752,1:POKE 87,0:POSITION 5,14:
? "By Mark Lewis Baldwin"
60 IF PEEK(CONSOL)<>7 THEN GOSUB 5500
70 IF STRIG(0)<>0 THEN 60
100 GRAPHICS 23:SETCOLOR 0,1,10:SETCOL
OR 1,6,8:SETCOLOR 2,12,6:SETCOLOR 4,0,
0:COLOR 1:FOR I=1 TO 40
110 PLOT 159*RND(0),95*RND(0):NEXT I
120 PLOT 74,42:DRAWTO 76,42:PLOT 84,42
:DRAWTO 86,42
130 PLOT 74,43:DRAWTO 76,43:PLOT 84,43
:DRAWTO 86,43
140 PLOT 74,44:DRAWTO 86,44:FOR I=45 T
O 51:PLOT 76,I:DRAWTO 84,I:NEXT I:PLOT
74,52:DRAWTO 86,52
150 PLOT 74,53:DRAWTO 76,53:PLOT 84,53
:DRAWTO 86,53
160 PLOT 74,54:DRAWTO 76,54:PLOT 84,54
:DRAWTO 86,54
170 FOR I=1 TO 4:MI(I)=0:NEXT I
180 AT=20:ALNT1=2+RND(0)*5:ALNP1=0:ALN
T2=RND(0)*AT:ALNP2=0:MS=4
190 TSPD=20:53=200:ESTP=5:ARNG=47:SC=0
:POKE 77,0
200 SC=SC+1:GOSUB 5310:T=T+1:IF T>TSPD
THEN AT=AT-1:T=0:IF AT<1 THEN AT=1
205 GOSUB 2000:IF FF=1 AND BASE<>0 THE
N 250
210 IF BASE=0 THEN 225
215 IF P=0 OR P=BASE THEN 300
220 COLOR 0:GOSUB 3000:BASE=0:GOTO 300
225 IF P=0 THEN 300
230 COLOR 1:BASE=P:GOSUB 3000:GOTO 300
250 FF=2:E=E+1:IF E>ESTP THEN E=0:RNGF
=1
260 R=47:HF=1:IF ABS(ALNP1)=BASE THEN
HF=2:R=ARNG-3:AF=1:GOTO 280
270 IF ABS(ALNP2)=BASE THEN HF=2:R=ARN

```



```

B-3:AF=2
280 IF MI(BASE)>6 THEN HF=3:R=MI(BASE)
290 GOSUB 1000:IF STRIG(TRIG)<>0 THEN
FF=0
300 Q=Q+1:IF Q>4 THEN GOTO 400
310 IF MI(Q)=0 THEN 200
315 MI(Q)=MI(Q)-MS:IF MI(Q)<2 THEN 150
0
320 ON Q GOTO 330,340,350,360
330 COLOR 0:PLOT 80,48-MI(Q)-MS:COLOR
3:PLOT 80,48-MI(Q):GOTO 200
340 COLOR 0:PLOT 80+MI(Q)+MS,48:COLOR
3:PLOT 80+MI(Q),48:GOTO 200
350 COLOR 0:PLOT 80,48+MI(Q)+MS:COLOR
3:PLOT 80,48+MI(Q):GOTO 200
360 COLOR 0:PLOT 80-MI(Q)-MS,48:COLOR
3:PLOT 80-MI(Q),48:GOTO 200
400 IF RNGF=1 THEN IF ALNP1=0 AND ALNP
2=0 AND ARNG>10 THEN ARNG=ARNG-2:RNGF=
0
410 IF Q>5 THEN Q=0:GOTO 600
500 ALNT1=ALNT1-1:IF ALNT1>0 THEN 200
530 IF ALNP1>0 THEN 570
540 IF ALNP1<0 THEN COLOR 0:GOSUB 4000
:ALNP1=0:ALNT1=INT(RND(0)*AT)+1:GOTO 2
00
550 ALNP1=INT(RND(0)*4+1):ALNT1=AT:IF
MI(ALNP1)<>0 OR ALNP2=ALNP1 THEN ALNP1
=0:ALNT1=0:GOTO 200
560 COLOR 3:GOSUB 4000:IF RND(0)*AT>0.
5 THEN 200
570 MI(ALNP1)=ARNG-4:GOSUB 5100:ALNT1=
AT:ALNP1=-ALNP1:COLOR 3:ON -ALNP1 GOTO
700,710,720,730

```

```

600 ALNT2=ALNT2-1:IF ALNT2>0 THEN 200
630 IF ALNP2>0 THEN 670
640 IF ALNP2<0 THEN COLOR 0:GOSUB 4010
:ALNP2=0:ALNT2=INT(RND(0)*AT)+1:GOTO 2
00
650 ALNP2=INT(RND(0)*4+1):ALNT2=AT:IF
MI(ALNP2)<>0 OR ALNP2=ABS(ALNP1) THEN
ALNP2=0:ALNT2=0:GOTO 200
660 COLOR 3:GOSUB 4010:IF RND(0)*AT>0.
5 THEN 200
670 MI(ALNP2)=ARNG-4:GOSUB 5100:ALNT2=
AT:ALNP2=-ALNP2:COLOR 3:ON -ALNP2 GOTO
700,710,720,730
700 PLOT 80,52-ARNG:GOTO 200
710 PLOT 76+ARNG,48:GOTO 200
720 PLOT 80,44+ARNG:GOTO 200
730 PLOT 84-ARNG,48:GOTO 200
1000 COLOR 2:ON BASE GOTO 1010,1020,10
30,1040
1010 PLOT 80,41:DRAWTO 80,48-R:GOSUB 5
000:COLOR 0:PLOT 80,41:DRAWTO 80,48-R:
GOTO 1050
1020 PLOT 87,48:DRAWTO 80+R,48:GOSUB 5
000:COLOR 0:PLOT 87,48:DRAWTO 80+R,48:
GOTO 1050
1030 PLOT 80,55:DRAWTO 80,48+R:GOSUB 5
000:COLOR 0:PLOT 80,55:DRAWTO 80,48+R:
GOTO 1050
1040 PLOT 73,48:DRAWTO 80-R,48:GOSUB 5
000:COLOR 0:PLOT 73,48:DRAWTO 80-R,48

```

Check type of hit.

```

1050 ON HF GOTO 1060,1080,1070

```

Missed.

1060 RETURN

Hit alien missile.

1070 MI(BASE)=0:SC=SC+5:GOSUB 5200:RETURN

Hit alien ship.

1080 SETCOLOR 4,0,14:GOSUB 5300:COLOR 0:SC=SC+50
1090 IF AF=1 THEN GOSUB 4000:SETCOLOR 4,0,0:ALNP1=0:ALNT1=AT:RETURN
1100 GOSUB 4010:SETCOLOR 4,0,0:ALNP2=0:ALNT2=AT:RETURN

Blow up base.

1500 SETCOLOR 1,0,14:COLOR 2:FOR I=0 TO 12:A=80-I:B=80+I:C=48-I:D=48+I:SETCOLOR 4,0,14*BC:BC=1-BC:SOUND 0,6,0,I+3
1510 SOUND 1,21,0,I+3:SOUND 2,27,0,I+3:SOUND 3,40,0,I+3:PLOT A,C:DRAWTO B,C:DRAWTO B,D:DRAWTO A,D:DRAWTO A,C:NEXT I
1515 COLOR 0:FOR I=15 TO 0 STEP -1:A=B0-I:B=80+I:C=48-I:D=48+I:SOUND 0,2,0,I:SOUND 1,15,0,I:SOUND 2,20,0,I
1520 SOUND 3,30,0,I:PLOT A,C:DRAWTO B,C:DRAWTO B,D:DRAWTO A,D:DRAWTO A,C:FOR J=1 TO 20:NEXT J:NEXT I

Display score.

1530 GRAPHICS 18:POSITION 2,2:?"#6;"your base was destroyed!!!":POSITION 2,5:?"#6;"YOUR SCORE WAS"
1540 POSITION 5,6:?"#6;SC:POSITION 1,8:?"#6;"THE TOP SCORE WAS":POSITION 5,9:?"#6;SCT
1550 IF SC>SCT THEN SCT=SC

Check console to change game modes.

1560 IF PEEK(CONSOL)<>7 THEN GOSUB 5500

Check to start a new game.

1570 IF STRIG(0)<>0 THEN 1560
1580 GOTO 100

Subroutine to check joystick.

2000 IF STRIG(TRIG)=0 THEN 2100
2005 ST=STICK(0):IF ST=11 THEN P=4:GOTO 2035
2010 IF ST=14 THEN P=1:GOTO 2035
2020 IF ST=13 THEN P=3:GOTO 2035
2030 IF ST=7 THEN P=2:GOTO 2035
2035 IF FF=2 THEN FF=0
2040 RETURN
2100 IF FF=0 THEN FF=1
2110 RETURN

Subroutine to draw or erase laser cannon.

3000 ON BASE GOTO 3010,3020,3030,3040
3010 PLOT 80,42:PLOT 79,43:DRAWTO 81,43:RETURN
3020 PLOT 86,48:PLOT 85,47:DRAWTO 85,49:RETURN
3030 PLOT 80,54:PLOT 81,53:DRAWTO 79,53:RETURN
3040 PLOT 74,48:PLOT 75,49:DRAWTO 75,47:RETURN

Subroutine to draw or erase alien ships.

4000 ON ABS(ALNP1) GOTO 4100,4200,4300,4400
4010 ON ABS(ALNP2) GOTO 4100,4200,4300,4400
4100 PLOT 78,48-ARNG:DRAWTO 82,48-ARNG:PLOT 82,49-ARNG:PLOT 80,49-ARNG:PLOT 78,49-ARNG:PLOT 80,50-ARNG
4110 PLOT 79,51-ARNG:DRAWTO 81,51-ARNG:RETURN
4200 PLOT 80+ARNG,46:DRAWTO 80+ARNG,50

:PLOT 79+ARNG,50:PLOT 79+ARNG,48:PLOT 79+ARNG,46:PLOT 78+ARNG,48
4210 PLOT 77+ARNG,47:DRAWTO 77+ARNG,49:RETURN
4300 PLOT 78,48+ARNG:DRAWTO 82,48+ARNG:PLOT 78,47+ARNG:PLOT 80,47+ARNG:PLOT 82,47+ARNG:PLOT 80,46+ARNG
4310 PLOT 79,45+ARNG:DRAWTO 81,45+ARNG:RETURN
4400 PLOT 80-ARNG,46:DRAWTO 80-ARNG,50:PLOT 81-ARNG,50:PLOT 81-ARNG,48:PLOT 81-ARNG,46:PLOT 82-ARNG,48
4410 PLOT 83-ARNG,49:DRAWTO 83-ARNG,47:RETURN

Laser fire sound routine.

5000 FOR II=10 TO 30:SOUND 0,II,10,10:NEXT II:SOUND 0,0,0,0:RETURN

Missile fire sound routine.

5100 FOR II=15 TO 0 STEP -1:SOUND 1,50-2*II,10,II:NEXT II:SOUND 0,0,0,0:RETURN
5200 SOUND 3,200,12,14:FOR II=1 TO 10:NEXT II:SOUND 3,0,0,0:RETURN

Start explosion for explosion subroutine.

5300 S3=40

Entry for continuation of the explosion.

5310 S3=S3+5:IF S3>120 THEN SOUND 2,0,0,0:RETURN
5320 SOUND 2,S3,8,(150-S3)/10:RETURN

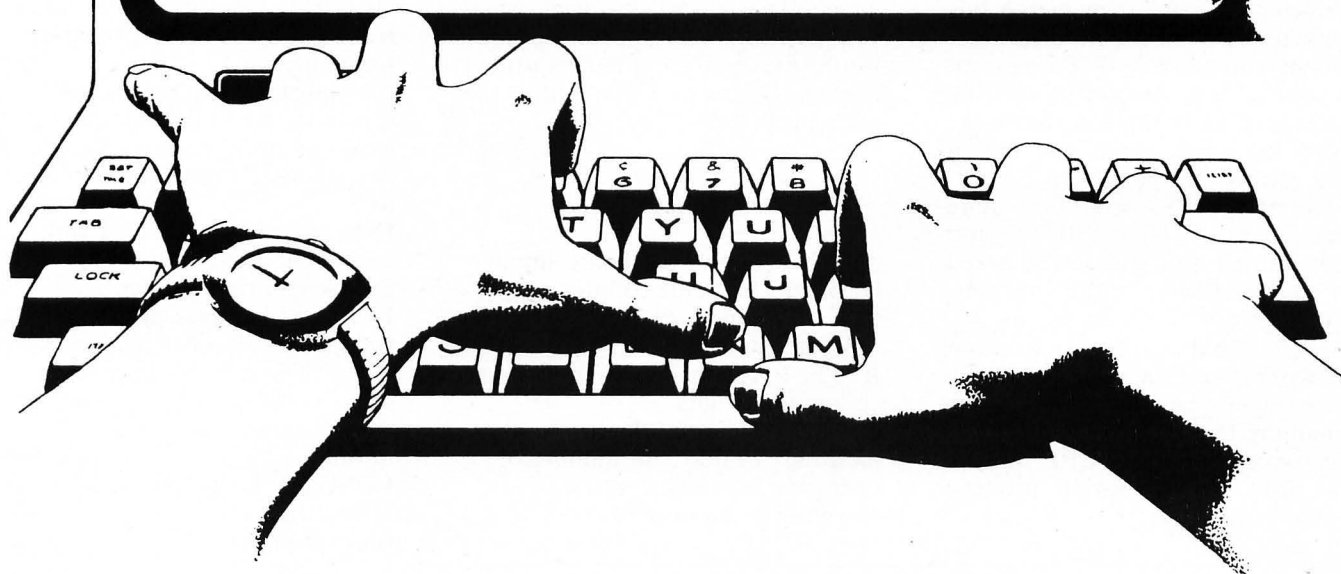
Subroutine to change between player modes.

5500 TRIG=1-TRIG:FOR ITR=0 TO TRIG:SOUND 0,100,10,8:FOR II=1 TO 20:NEXT II:SOUND 0,0,0,0:FOR II=1 TO 20:NEXT II
5510 NEXT ITR:RETURN



ATARI®

Atari Banner Machine



by Alan J. Zett

ATARI Banner Machine is a banner printing program for the ATARI® 400/800 with 16K RAM for tape, 24K for disk, and a printer. Media subscribers will receive an enhanced version with a redefined character set.

To go along with the general theme of word processing for this month's issue, I was asked to write a program that would produce printed banners for the ATARI®. We had already selected similar programs for the Apple and TRS-80®, but could find none for the ATARI® among our submissions.

At first I thought of writing a translation, but discarded this idea

when a different thought suddenly leaped into my mind: Since the ATARI® character set uses a map of 8 by 8 bits to form a character, why not use this same map to create characters on a printer as well as on the screen? In my *Character Generator* program in the October, 1981, issue of *SoftSide*, I had already done a lot of the work necessary to create such over-sized characters from the ATARI's internal character set. So by going one step further, I developed a version of *Character Generator* that would print the ATARI® character set, on its side, enlarged to any degree. The characters printed would be an exact duplicate of the characters as displayed on the screen. Taking this basic idea and adding many options has led to *Banner Machine*.

Banner Machine is designed with *Character Generator* in mind. The two work hand-in-hand. Note that the print quality of *Banner Machine* is totally dependent on the character set in memory. This is why I recommend that you use *Character Generator* first to define a new character set, and then merge this character set with *Banner Machine*. This has been done to some extent for those who subscribe to the media version of *SoftSide*; however, I still recommend that you design your own to suit your needs.

The program's operation is quite straightforward. Type "RUN", and after a small wait you will be asked for the character width of your printer. Normal responses are 40, 80, or 132, but anything in between will work if necessary. If you choose 40 characters per line, most of the remaining options will be disabled. This is because at 40 CPL, only a few print options would produce legible print.

The height and width questions allow you to tailor the physical dimensions of the characters to your own taste. The text positioning question helps to make room for other things that you may want to add when the banner is done. You are also given the option of using any ASCII or keyboard character to do the actual printing. This is followed by a routine to allow you to set up your printer with special control codes if desired. Just enter the ASCII values of the control characters, one after the other, and conclude by entering "999". The only option after this is to press the spacebar to stop printing. (The spacebar will only be acknowledged AFTER a character has finished printing. You may press the spacebar ahead of time and the ATARI® will remember.)

Well, that about wraps it up. I hope you get as much use out of this program as I have. One final note: I sometimes have to remind myself that printer ribbons and print heads wear out quite fast when you go through 2000 sheets of printer paper. That's what happens when you foolishly ask someone if they would like a free computer-printed banner. I hope you don't have the same problem!

Variables

A: Miscellaneous keyboard input.
 B: Binary bit value counter.
 BIT: Bit number (0-7) in a byte of character data.
 BYTE: Byte number (0-7) of character set data.
 C: In line 50, print density modifier. In line 370, number of character in message string currently being printed.

CH: Character height. Calculated width of a bit, in characters, when printed.

CH\$: Print characters. Defaults to a space for a zero bit in a normal character, a pound sign (#) for a one bit in a normal character, and vice versa for inverse characters.

CH(7,7): An 8 by 8 array of bit mapped character data. (See *Take Apart: ATARI® Quest*, October, 1981, *SoftSide*). Bit set = 1, bit reset = 0.

CHR: In line 370, ASCII value of character number "C" in message. In line 380, adjusted value for character data search location.

CPL: Character width of printer. 40, 80, or 132 characters per line (CPL).

CPOS: Memory location of the first byte in a letter's character set data.

CW: Character width. Calculated height of a bit, in lines, when printed.

FLAG: Inverse video flag. Used to reverse printing for an inverse character. Normal = 0, inverse = 1.

HEIGHT: Print height. A value from 1 to 9 (10 to 90 percent of full-size print.)

IN: Indent value. Number of spaces to indent printed characters based on print mode and print size descriptions. Preset for 40 CPL to 9.

IN\$: Indent string. Filled with spaces and used with IN to produce a printed indentation.
 LC: Last character printed. Allows program to skip calculation of bits if next character to print is the same as the last one. Inverse bit data is considered the same as normal.

LENGTH: Print length. A value from 1 to 9 (10 to 90 percent of full-size print).

MSG\$: Holds entire banner message. Dimensioned to the amount of free memory divided by four, to allow a large message.

PM: Print mode. Text positioning information. Except for line 180, flush left = 0, centered = 1, flush right = 2.

X, Y: Miscellaneous FOR/NEXT loops.

Z: Temporary storage for character set data bytes.

Continental Adventures

Continental Adventures presents three adventures and one graphics game for the Atari 400 and 800 computer owner

The Ghost Tower — Combat with diabolical demons, 16K \$16.95
Town of Derango — Avenging the death of a father, 8K \$16.95
Talisman of Power — A search for the four keys of Gremlock, 16K... \$18.95
Super Shape Builder — A graphics game for creating your own pictures. Joysticks reqd. 8K \$14.95

4975 Brookdale Dept. 06
 Bloomfield Hills, Mich. 48013
 (313) 645-2140

330 IF A=78 THEN 360

Turn on cursor and print escape message.

340 POKE 752,0:?: "ENTER 999 TO END
...":?

Get printer control code. Check whether it corresponds to the menu's escape code; if not, print to printer and loop for more.

350 INPUT A:IF A<>999 THEN LPRINT CHR\$(A);:GOTO 350

Print fourth menu heading, turn on cursor, and input message in MSG\$. Turn off cursor and set last character printed to 999.

360 ? "ENTER BANNER MESSAGE:";POSITIO
N 2,11:?"MESSAGE";:POKE 752,0:INPUT M
SG\$:POKE 752,1:LC=999

Set up "C" to loop through characters in message (MSG\$). Get ASCII value of character "C" into CHR and reset inverse video flag. Check if character is inverse; if so, set flag and adjust the value of CHR.

370 FOR C=1 TO LEN(MSG\$):CHR=ASC(MSG\$(C,C)):FLAG=0:IF CHR>127 THEN FLAG=1:CHR=CHR-128

Modify character value according to the mixed-up way that ROM holds character set data.

380 IF CHR<96 THEN CHR=CHR-32*(CHR>31)
+64*(CHR<32)

Get the memory position of the first byte of character data into CPOS.

390 CPOS=CHR*8+(PEEK(756)*256)

Print printer processing display heading, total banner message, and portion of message printed.

400 ? "YOUR BANNER MESSAGE":?:? MSG
\$:?:? "PART PRINTED SO FAR":?:? MSG
\$(1,C)

Print expanded visual display heading and check if the last character is the same as the next one. If so, skip bit processing routine.

410 ??: "WORKING ON LETTER":?:? :IF LC
=CHR THEN 490

Bit processing routine. Takes all bytes of a letter's character data and breaks them down into 64 bits which are then bit mapped and stored in an 8 by 8 array in CH(n,n).

420 FOR X=0 TO 7:Y=-1:B=256
430 Z=PEEK(CPOS+X)
440 B=B/2:Y=Y+1:IF Y>7 THEN 480
450 IF Z-B<0 THEN CH(X,Y)=1:GOTO 470
460 IF Z-B>=0 THEN CH(X,Y)=2:Z=Z-B
470 GOTO 440
480 NEXT X

Plot character bit map with normal and inverse spaces to produce an expanded form of the character currently under processing. Also reset ATARI's attract mode flag.

490 FOR X=0 TO 7:FOR Y=0 TO 7
500 IF FLAG=0 AND CH(X,Y)=1 OR CH(X,Y)
=2 AND FLAG=1 THEN ? " ";
510 IF FLAG=0 AND CH(X,Y)=2 OR CH(X,Y)
=1 AND FLAG=1 THEN ? " ";
520 NEXT Y:?:NEXT X:POKE 77,0

Prompt for halting of banner print, and erase any keyboard input by nulling out the "last-key-pressed" memory location.

530 POSITION 5,23:?"PRESS SPACEBAR TO
STOP PRINTING"::POKE 764,255

Set BIT to cycle through all eight bits of character data. Set up X to print the correct number of lines for a given character width.

540 FOR BIT=0 TO 7:FOR X=1 TO CW

If flush left was not chosen, print the leading indent.

550 IF PM<>0 THEN LPRINT IN\$(1,IN);

Set BYTE to cycle through all eight bytes of character data.

560 FOR BYTE=7 TO 0 STEP -1

Set up Y to print the correct number of characters for a given character height.

570 FOR Y=1 TO CH

Print proper character for any bit and flag combination.

580 LPRINT CH\$(CH(BYTE,BIT)+FLAG,CH(BY
TE,BIT)+FLAG);:NEXT Y

Do all bytes, advance paper to next line and finish character.

590 NEXT BYTE:LPRINT :NEXT X:NEXT BIT

If no key has been pressed, then jump to line 630.

600 IF PEEK(764)=255 THEN 630

Otherwise get the character into A and erase character from memory.

610 GET #1,A:POKE 764,255

If the key was the spacebar, then jump to line 640.

620 IF A=32 THEN 640

Continue printing banner with the next letter in MSG\$.

630 LC=CHR:NEXT C:GOTO 660

Make sure operator wants to stop printing. If reply is "Y" then POP the "C" FOR/NEXT variable and jump to line 670.

640 ? "ARE YOU SURE YOU WANT TO QUIT?
(Y/N)":GET #1,A:IF A=89 THEN POP :GOT
O 670

Otherwise, jump to line 630.

650 GOTO 630

Print completion message.

660 ? "ALL DONE!"

Prompt for another banner.

670 ??: "WANT TO MAKE ANOTHER? (Y/N)"
:GET #1,A

If reply was "N" then say goodbye, turn on cursor and END program.

680 IF A=78 THEN ? "OK! BYE!":POKE 75
2,0:END

Check if next banner is to use the same parameters as the last. If reply is "N", then rerun program.

690 ??: "USE SAME PARAMETERS? (Y/N)"
:GET #1,A:IF A=78 THEN RUN

Otherwise, jump to 360 for next message.

700 GOTO 360

Word Processing Programs for the ATARI®

A review by Sheldon Leemon

Text Wizard

from Datasoft, Inc. Requires 32K. Suggested retail price: \$99.95.

Letter Perfect

from LKJ Enterprises. Requires 24K. Suggested retail price; \$149.95.

Word Processor

from Atari, Inc. Requires 48K. Suggested retail price: \$149.95.

One of the nice things about owning a personal computer is that new applications always seem to be turning up. An application which seems to have gained sudden recognition among microcomputer owners is word processing.

Word processing software allows a person to prepare a document by typing the words onto the computer display screen, making all editing corrections on the screen, and then using a printer to print the document on paper. Regular computer disk storage is used for preserving the document on file for later use.

This method of document preparation provides a dramatic increase in ease and efficiency over the old method of composing a rough draft at the typewriter, proof-reading the draft, and then typing the finished copy over again. Who in his right mind would opt for repeated entry of every keystroke necessary to prepare a document, when offered the alternative of typing the document once, and entering only those changes which are necessary?

Part of the reason for the sudden rise in interest in word processing software is the recent appearance of relatively inexpensive high quality dot-matrix printers. The popularity of printers like the Atari 850/Centronics 737 and Epson MX-80 is such that although ATARI® computers are being marketed primarily as consumer products, there are now three popular programs for pursuing this more serious application on the ATARI®. These programs are *Text Wizard*, *Letter*

Perfect, and Atari's own *Word Processor*. Each program will work with the two brands of printers mentioned above, although certain features are implemented for only one printer or the other, but not both.

All of these programs have in common two major functions. The first is to allow the user to set down thoughts in words, and then to make changes for the purpose of editing content, as well as correcting grammar and spelling. The second function is to establish the format in which those words will appear on paper. This includes such matters as margin settings, spacing, and indentation. Other functions include sav-

"The method of choosing which of the major or minor program functions will be performed varies significantly between these programs."

ing text to a disk file, and loading in a previously-prepared text. In general, *Text Wizard* and *Letter Perfect* handle these tasks in a similar fashion, while a somewhat different approach is taken by *Word Processor*.

The method of choosing which of the major or minor program functions will be performed varies significantly between these programs. *Text Wizard* is completely command driven. This means that it does not prompt the user with menu selections to change functions, but rather requires that the user enter a certain combination of keystrokes to make the change. *Word Processor*, at the other extreme, has

several menu levels that help guide the user through every decision. *Letter Perfect* takes the intermediate position, offering one main menu that allows switching between major functions, but using keystroke commands for editing and print formatting functions.

Which approach is preferable is mostly a matter of personal taste. Some people like the convenience of menu prompts, as opposed to having to learn a number of command sequences. Others find that once they have used the program for a while, the menu prompts become irritating, and prefer to learn a few commands which will get them instant results.

Screen Display

Another example which illustrates the differences between the systems employed in the various programs is the way in which the screen display is handled for text editing. The problem presented is that the ATARI® screen display is limited to 40 characters per line, while the printed line may contain up to 132 condensed-print characters.

In the case of *Text Wizard* and *Letter Perfect*, the approach is to let the computer worry about where to start a new line, and to concentrate on keeping the screen display readable, rather than having it correspond exactly to what is being printed. To that end, both of these programs use a wrap-around technique, which consists of moving any word that would be split at the end of a line to the beginning of the next line.

Word Processor, on the other hand, uses a "moving window" approach. This looks on the screen as a window on a larger area that includes the entire line of print. A row and column marker in the upper right hand corners lets you know the current position of the cursor. When you get past 40 columns on the screen, the display scrolls to the left, simulating movement of the

K-Byters

ANOTHER PROGRAMMING CHALLENGE

Some time ago *SoftSide* began inviting its readers to submit "One Liners" — self-contained single-line programs for the TRS-80®, Apple, or ATARI® which would provide a continuously changing graphics display. The response has been excellent, and we're still looking for more submissions.

Now we have a new challenge for you as well: "K-Byters." A K-Byter is a BASIC program which fits into 1K (1024) bytes of program memory. There aren't any restrictions on the nature of the program, other than its size. It can be a graphics display, a game, a mini-adventure, or anything your imagination and programming skills can create.

Note that the program does not have to RUN in 1K of memory; it can use as much RAM for arrays, strings, graphics mapping, etc., as you need. We'd prefer that it be able to run in a 16K system, but this is not an absolute limit.

Here then are the official rules:

1. The program must be written for the Apple, TRS-80®, or ATARI®, entirely in BASIC (although it may create and call Machine Language routines).
2. The program must occupy no more than 1024 bytes of memory before running.
3. The program must be submitted on tape or disk, accompanied by your name, address, phone number, and a brief written description of its operation.
4. The tape or disk will be returned only if accompanied by a self-addressed envelope with adequate postage AFFIXED (do not send money).
5. Winners will have their programs published in *SoftSide* and will receive a \$10 software certificate for their programming excellence!

Send submissions to:

K-Byters, c/o *SoftSide*
6 South Street
Milford, NH 03055

ATARI®

window to the right. This allows you to see the position of each word on the screen as it will actually appear on the printed page.

Obviously, such an approach will be advantageous when it is important to know the exact layout of the page before it is printed. One such case would be where the user is trying to set up information in a columnar format. This task is difficult for those using *Letter Perfect*, and even more difficult for users of *Text Wizard*, as that program does not even have tab stops which can be set by the user.

However, the window concept does have its drawbacks, as only a portion of each sentence is displayed on screen at a time. For those applications where the exact layout is not critical, the ability to reread previous sentences without having to scroll the screen may be more valuable than being able to determine the exact position of each word on the page.

Text Editing

The actual process of text editing is very similar to the method of screen editing used when writing programs on the ATARI®. If there is something on screen which you wish to change, you use the editing keys to move the cursor to the correct spot, to insert lines or spaces, delete lines or spaces, or just type over the old text.

While normal ATARI® cursor placement is handled by the arrow keys, these programs feature other commands to help you get to the spot you wish to edit more quickly. All have commands that allow you to go to the top of the text or the bottom of the text. *Letter Perfect* and *Word Processor* also have commands that let you go to the beginning of the line or the end of the line. Each program allows you to scroll the text up or down, although the means of scrolling up is rather clumsily implemented in *Letter Perfect*.

Letter Perfect and *Word Processor* allow you to delete lines or insert blank lines much as is done in regular ATARI® editing. When you move the text around this way in *Letter Perfect*, the screen wrap-around is not preserved, and

must be restored with a separate command.

The method of inserting text is somewhat different in *Text Wizard*. Instead of inserting fixed-length blank lines, it has an Insert mode, which allows you to move the cursor to a spot and add text in that spot while the rest of the text is scrolled down. Screen wrap-around is preserved at all times, which keeps the appearance neat, but which makes some editing features work a bit slower than they normally would.

The Insert mode is an extremely convenient feature, and is also found on *Word Processor*, but not *Letter Perfect*. *Word Processor* also has a number of different options for deleting a specific word at a time, part of a line, an entire line, or a block of lines. Unlike the other two programs, it will allow you to restore a line that has been edited, back to its original state.

Advanced Editing Features

Advanced editing features are available in these programs which allow moving around and duplicating blocks of text, much like "cut and paste" editing on paper, only without the mess. This is accomplished by inserting a designated block of text into a buffer, and then transferring the contents of the buffer to a specific area of text. The buffer features of *Letter Perfect* and the *Word Processor* appear to be a bit more flexible than *Text Wizard*, which allows simple moves of short blocks of text rather easily, but does not allow more sophisticated operations such as combining blocks of text in a buffer, or holding text in the buffer for long periods of time.

Another advanced function that is supported in all three programs is the Search function, which moves the cursor to the each place in the text where a designated word or phrase occurs. This can be combined with an even more powerful Replace function, which will replace each occurrence of the word or phrase with another word or phrase. Such a function can be very valuable for bad spellers, for example, who consistently misspell the same word. In *Letter Perfect* and *Text Wizard*,

the user must verify each change (the cursor will move to the word, and you have to enter a keystroke before it will be replaced). *Word Processor* offers the choice of Replace upon verification, or without verification.

Formatting the Printout

The second major function which the user will perform with these programs is setting up the format in which the document is to be printed. Both *Letter Perfect* and *Text Wizard* have default formats which allow the document to be printed neatly without special effort by the user. The programs will automatically set margins, right-justify (insert spaces within the line so that each line ends exactly at the right margin), and will give a form-feed command to the printer at the end of the page, to advance the paper to the top of the next sheet.

Of course, there are many options that can be selected in place of the default values. This selection is accomplished by the use of what are called "format lines." These are lines of characters imbedded in the body of the text which give commands to the printer. Adjustments can be made such as setting left, right, top, and bottom margins; line spacing; turning off justification; or using one of the printer's alternate character fonts. These commands can be given as text is entered, or can be added afterward, allowing, for example, changing margins to allow for indentation of certain paragraphs.

In *Text Wizard*, right and left margins are set in dot widths, instead of character widths, which allows for very fine adjustment. Also, unlike the other two, *Text Wizard* permits line spacing to be set by half-spaces. One of the nice extended functions available in all three programs is the ability to designate page headings, as well as automatic page numbering at either the top or bottom of the sheet.

Print formatting is done a little differently on *Word Processor*. A menu option allows setting of margins, line spacing, etc. This can be done before entering text, and the result will be that the screen

display will be an exact representation of how the printed page will look. If you later decide to change the format, a menu option allows you to change the whole document, or any page. Since files are saved as a series of separate pages, there are also depagination and pagination commands to change the allocation of text on each page. This allows extremely flexible layout options, but takes a little more effort than the other systems, which leave paging and layout to the computer.

Finally, there is the actual printing. *Letter Perfect* has the most flexible system. It allows you to print single or multiple copies of any page or combination of pages. While *Word Processor* also allows these options, you cannot print a document straight through; the program pauses at the end of every page and prompts you to advance the paper to the top of the next sheet and type a carriage return. *Text Wizard* only allows you to print straight through from the beginning or middle of text to the end, and only will let you print one copy at a time.

Documentation

There are some other factors, not related to the software itself, that potential purchasers of these programs will want to consider. The first is the documentation. *Word Processor's* is by far the most elaborate. It consists of an instruction manual, a reference guide, system hook-up instructions, an audio instruction tape, and a diskette of sample material, all in a colorful vinyl binder. The manual assumes that you've never seen a computer, and takes you step by step from plugging it in to putting the paper in the printer. The others are on a less ambitious scale, but are clearly written and easy to follow.

One aspect of *Letter Perfect's* manual that is somewhat disappointing is that it contains a fairly large number of typographical errors. Although the program itself has been revised, the body of the manual remains the same as for the earlier version. A single update sheet at the beginning explains the

new features which are available.

Text Wizard's manual takes the user through a sample editing session step by step. This approach is very reassuring to the novice, and is a fine teaching tool. However, the manual does leave some questions unanswered. For example, it doesn't fully explain the function and proper structure of the text-formatting lines. Of all of the complaints I have heard about *Text Wizard*, most have involved an inability to use one or more printing functions as described, and most of these failures resulted from improper placement of the commands in the text-formatting lines.

Interfacing Programs

Another area of interest to the user is interfacing this program with



duplicating service

307 West Main Street
Maple Shade, NJ 08052

(609) 667-1667

- AMP "Data-sette" blank cassettes for digital use
- Cassette Storage Boxes
- Cassette Labels - Custom printing & blank
- Custom Record Album production from your tapes
- Stereo and Spoken Word cassette duplication

Call or write to:

Jerry

for more information.

All cassette work at
AMP R. & D. is custom work
to fit your needs.

PROGRAM FEATURES

	Word Processor	Text Wizard	Letter Perfect
Cursor control (one space up, down, right, left, begin line, end line, last page, next page, top of text, bottom of text)	XXX	all but begin/ end line	XXX
Scrolling text window	XXX		
Delete word	XXX		
Restore deleted text	XXX		
Auto-insert	XXX	XXX	
Move text	XXX	XXX	XXX
Duplicate text	XXX	XXX	XXX
String search	XXX	XXX	XXX
Search and Replace	XXX	on verify only	on verify only
Print margin control (top, bottom right, left)	XXX	XXX	XXX
Subscripts, Superscripts, Underlining	825	825	MX-80 + Graftrax
Right justify	not PS set	XXX	XXX
Double-column printing	825	825	825
Headers and footers	headers	XXX	XXX
Page numbering	XXX	XXX	XXX
Database merge			XXX
Output control codes	XXX		XXX
Print selected pages	XXX		XXX
Print multiple-page copies	XXX		XXX

others. For example, *Letter Perfect* has mail-merge capabilities with its sister program, *Data Perfect*. This means that mailing lists or other databases created with *Data Perfect* can be merged with *Letter Perfect* form letters, so that the computer will literally fill in the blanks in these form letters with data provided by files created with the database.

Datasoft is working on a revised *Text Wizard* that will allow merger with files created by *File Manager 800*, an excellent database program by Synapse Software. This revised version, which should be available in this month or shortly after, will also add some features that take advantage of the Graftrax ROMs for the Epson printers, such as underlining and subscripting. These features are already implemented in the revised version of *Letter Perfect*.

Word Processor has no capabilities for merging data from separate data files into text.

In the same vein, the prospective user should consider that *Letter Perfect* files are not DOS compatible. This means that it cannot handle some of the extended applications that the other programs permit, such as using the Search and Replace function in editing BASIC programs, or preparing text files for uploading via modem to a computer bulletin board.

From the detailed comparisons given above, a general picture begins to emerge of the differences between these programs. *Word Processor* is very menu-oriented and page-oriented. It allows the most versatility and accuracy in page layout, at the expense of some convenience. Although it is the only program of the three that will not

right-justify the proportional type of the Atari 850/Centronics 737, this is offset to some extent by the accurate representation of page layouts which allows the user to hyphenate words as if using a typewriter. The biggest failing of *Word Processor* may be its inability to type a multipage document straight through.

Letter Perfect has nice print options, allows the user to output command codes directly to the printer, but is somewhat lacking in its user-interface. For example, the program title that always occupies the top line of the editing screen is distracting. And the multitude of control commands are a little difficult to remember. Its files not being DOS compatible precludes its use for some functions, as outlined above.

Text Wizard, although slightly less versatile than the other two, is extremely easy to use. The command codes are well thought out, and are logical extensions of the computer's own system of editing commands. Such features as the screen border changing colors to remind you when you are in Insert mode, or when you are running out of screen memory, lend a nice touch. Its features are sufficiently powerful for most applications, and its ease of use would probably make it the right choice for applications that did not require special page layouts or multiple column printing. The fact that it costs less than the others doesn't hurt, either.

While the above programs pose no serious threat to the sophisticated and expensive stand-alone word processors used by businesses, they are all good enough to make you want to throw away your typewriter. Any ATARI® owner who has an 80-column printer should seriously consider the purchase of one of these programs as a relatively painless way of greatly expanding the utility of his or her system. You might as well become familiar with word processing now, because as the price-to-performance ratio of computers and printers continues to rise, and software packages such as these add features to rival the expensive stand-alone systems, there is every possibility that microcomputer word processing systems will replace the typewriter entirely. ☺

K-Byter

Micro Adventure

A TRS-80® K-Byter by J. Joseph Felten, Corydon, IN

The goal of this real-time graphics adventure is quite simple: to survive. You are being pursued through a dungeon by a deadly dragon, and you must move yourself around using the arrow keys. Among the many obstacles which are scattered around the screen, there are treasures which you are trying to collect along the way. When you've picked up as many of these as you dare, you attempt to make your escape through the door of the dungeon.

You are displayed on the screen as an "O", the dragon as a "#". Obstacles are shown as irregular graphics blocks, and the door to safety is a solid rectangular block near the top center of the screen. An "=" shows the location of a staircase leading downward. (The obstacles increase as you descend.) In addition, one of the obstacles will occasionally change into a magical sword (shown by an arrow) which is capable of killing the dragon.

The author wishes to acknowledge Steve Geswein as the author of the original Pet version of this adventure.

```

1 *MICRO ADVENTURE
  BY J. J. FELTEN
5 CLS:INPUT"LEVEL";S
10 CLS:P=15360:A=64:FORI=1TORND(S*20):POKERND(1023)+P,173:NEXT:P
   OKE15456,188:X=2:Y=32:W=RND(15):Z=RND(63):POKEP+A*X+Y,79
20 FORI=1TO5:POKERND(703)+15680,42:NEXT
  
```

```

30 POKERND(960)+15423,61:S0=RND(960)+15423:POKES0,93:POKEP+A*W+Z
  ,35
100 J=PEEK(14400):IFJ=0THEN200
105 IFRND(0)<.05ANDM=0THENPOKES0,93
110 U=0:N=-1:IFRND(0)<.05THEN200
115 IFJANDA,N=Y+1ELSEIFJAND32N=Y-1ELSEIFJAND16N=X+1:U=1ELSEIFJAN
  DBN=X-1:U=1
135 IFUANDN<0THENM=0
140 IFUANDN>15THENM=15
145 Q=P+A*X+N:IFUTHEM=Q:P+A*N+Y
147 E=PEEK(Q)
150 IFE=42T=T+1ELSEIFE=188THEN1000ELSEIFE=173THEN200ELSEIFE=93M=
  1ELSEIFE=35THEN1100ELSEIFE=61S=S+1:H=0:M=0:GOTO10
165 POKEP+A*X+Y,32:POKEQ,79:IFUTHEM=N:ELSEY=N
180 IFY<1THENX=X-1:Y=Y+A
190 IFY>ATHENX=X+1:Y=Y-A
200 IFHTHEN100
210 IFRND(0)<.3THENPOKES0,173
220 IFRND(0)<.1THEN100
230 X1=W-X:Y1=Z-Y:IFX1=0ANDY1=0THEN1100
240 IFABS(X1)>ABS(Y1)THENM=W-SGN(X1):U=1:ELSEN=Z-SGN(Y1):U=0
250 POKEP+A*W+Z,32:IFUTHEM=N:GOTO290
270 Z=N
280 IFZ<1THENM=W-1:Z=Z+A
285 IFZ>ATHENM=W+1:Z=Z-A
290 POKEP+A*W+Z,35:GOTO100
1000 PRINT@19,"YOU WIN WITH"TRASURES":END
1100 IFM=0THEN1150
1110 IFRND(0)>.9THENPRINT@24,"THE SWORD BROKE":PRINT:GOTO1150
1130 POKEP+A*W+Z,33:POKEP+A*W+Z,32:H=1:GOTO100
1150 PRINT@82,"YOU LOSE WITH"TRASURES":FORI=1TO2000:NEXT:RUN
  
```

by Stephen Milliken

Screen Print is a routine that works with Level II, Disk BASIC, TRSDOS, NEWDOS and probably any other TRS-80® operating system. It is included as a bonus program on this month's TRS-80® DV.

One of the first things most people look for when they get a printer is some method to enable them to get a screen listing. Some printer manuals give simple BASIC routines to do this, but they are slow and cumbersome, having to be included in a BASIC program.

NEWDOS, with its JKL function, satisfies this need somewhat, but even here things don't seem right. As it is, the JKL function does not support graphics, and although this can be corrected with various ZAPS or POKES, there is still something wrong. The "something wrong" is that 16 lines on the printer just don't look like 16

lines on the screen. The aspect ratio, the ratio of a picture's width to its height, is not right.

Here is a BASIC program that will correct this problem by POKeing a relocatable Machine Language program into memory which will give you a real-time "extended graphics" screen print function. This function, activated by pressing the SHIFT and down-arrow keys, will print the contents of the screen on 32 printer lines. This corrects the aspect ratio and the printer listing is now approximately the same size as the screen.

This is made possible by using a technique I call "extended graphics." What this means is that for each screen line, two printer lines are made. The first line includes all non-graphics characters and the top of the "extended graphics." The second line contains just the bottom of the "extended graphics." Try this

function with any sketch or graphics program (i.e., *Old Glory*, *SoftSide*, June, 1981), and you will appreciate it.

The BASIC program when RUN, will check MEM SIZE and adjust it to accommodate the Machine Language code. The Machine Language code is relocated using the address figured out above as a base and is POKed into the now-protected memory. Because of this, no MEM SIZE has to be set unless some other code is to be used also (i.e., the sound routine in "COMMANDing BASIC", *SoftSide*, November, 1981). The one exception is if the user will be jumping back and forth between Disk BASIC and the disk operating system, see below.

How to use *Screen Print*:

All: Load any other non-relocating Machine Language

SCREEN



routines first. This routine will load below them.

Level II:

- 1) RUN program.
- 2) Shift-down-arrow activates, BREAK aborts.

TRSDOS:

A) If only Disk BASIC to be used, then same as Level II.

B) If jumping back and forth between Disk BASIC and TRSDOS:

1) Set MEM SIZE 100 less than top of memory. *

2) RUN program.

3) Each time you re-enter Disk BASIC answer "MEM SIZE?" with 200 less than number calculated in 1.

4) Shift-down-arrow activates, BREAK aborts.

NOTE: In TRSDOS, hitting BREAK sometimes causes strange things to happen. Typing DIR when you enter seems to cure this problem.

NEWDOS:

1) RUN program.

2) When entering Disk BASIC set MEM SIZE 200 less than end of memory. *

3) Shift-down-arrow activates, BREAK aborts.

NOTE: JKL function still works, so what I do is to use that for any listing without graphics, and use shift-down-arrow for any listing with graphics.

*If other Machine Language programs are loaded above this one, make sure you add appropriate number to this figure.

I have found that the program does work with the lower-case driver I have by:

1) RUNning the BASIC program first, following instructions above.

2) Running the lower-case driver.

3) For disk operating systems make sure you now save enough memory on re-entry into BASIC for both the driver and the screen print function.

The program is set for an MX-80 in non-TRS-80® mode. Here are some modifications:

A) If your printer uses standard TRS-80® code:

1) Change number in line 30 from 18232 to 17772.

2) Change lines 210 and 260 to:

210 DATA 0,0

260 DATA 0,0

B) If your printer does not support graphics characters:

1) Change number in line 30 from 18232 to 17974.

2) Change lines 210 and 260 to:

210 DATA 62,46

260 DATA 62,32



PRINT

GOthic LEtTER PRINtER

by Ronald M. Tutone

***Gothic Letter Printer* is a banner-making program for a TRS-80® with a minimum of 32K RAM and a 132-column printer.**

Gothic lettering originated in western Europe around the 12th through the 15th century. *Gothic Letter Printer* recreates this beautiful and outstanding lettering form.

To produce your sign, follow these two steps:

1. Have your printer set for condensed mode (132 characters per line).
2. Type in your message.

Valid gothic characters are the letters A-Z, numbers 0-9, and the following punctuation: exclamation point, question mark, period, colon, semicolon, comma, hyphen, left and right parentheses, double quotes, apostrophe, dollar sign, and ampersand.

Special note: Because of the very large amount of data used in this program, we suggest that before saving the program in its final form, you first type the following one liner

from the command mode (READY prompt):

```
RESTORE: CLS: A=0: FOR X=1
TO 1506: READ A$: A=A+LEN
(A$): NEXT X: IF A < > 16962
THEN PRINT "DATA BASE
ERROR!" ELSE PRINT "THE
DATA APPEAR TO BE O.K."
```

After you press ENTER, the computer will go blank for a while and then print a message. If you get the "DATA BASE ERROR!" message then there are some data in your program which are not the same as is listed in the magazine. If you get the "DATA APPEAR TO BE O.K." message then your data should be the same as listed in the magazine. However, it is possible for the data to appear correct, and still contain a problem. A test run of all valid characters will show for sure if there is anything wrong.

Variables

A\$(n): Contains one gothic letter in coded form. A\$(n) is decoded to create one gothic letter.
C\$: Character used to create the

sign. Value will be the "delete" character. C\$ may be changed by the user if desired to any alphanumeric character.

CT: Number of lines in one gothic letter.

DT\$: List of the valid gothic characters. It is used to check the user's message for invalid characters.

I, II, J: Counters.

K\$: Advance paper or restart program.

L\$: Get one character from user's input.

M\$\$: User's message.

NLT: Number of valid input characters. Value will be 50.

NU: Number of blanks or characters in part of one gothic letter line.

N1\$, N2\$: Used to derive NU from a string of numbers into numeric values.

QUOTE: Determine opening or closing quotation marks. Value will be 1 or 0.

S\$: Character from the user's message for decoding.

ST\$: One line of the gothic letter to be printed.

X: Position on screen where user input is read.

TRS-80[®]

```
*****
* TRS-80 Model I/III BASIC *
* "GOTHIC LETTER PRINTER" *
* AUTHOR: Ronald M. Tutone *
* (c) 1982 SoftSide *
*****
```

Reserve memory space and define variables.

```
100 CLS: CLEAR 1700: NLT=50
150 DIM D$(NLT), A$(60), PTR(256)
```

Display program introduction.

```
200 PRINT@335, CHR$(23); "INTRODUCING"
250 FOR I=1 TO 1800: NEXT: CLS: FOR I=1 TO 500: NEXT
300 PRINT@332, CHR$(23); "THE GOTHIC LETTER"
350 PRINT@469, CHR$(23); "PRINTER"
400 PRINT@585, CHR$(23); "BY RONALD M. TUTONE"
450 FOR I=1 TO NLT-1: READ D$(I): NEXT I: RESTORE
475 FOR I=1 TO 2000: NEXT I: CLS
```

Initialize variables.

```
550 QUOTE=0: C%=CHR$(127)
```

Display instructions.

```
600 PRINT: PRINT TAB(10); "G O T H I C   L E T T E R   P R I N T E R"
650 PRINT TAB(10); " * * * * *   * * * * *   * * * * * * * * * * "
700 PRINT@264, "DIRECTIONS FOR CREATING A BEAUTIFUL GOTHIC SIGN:"
750 PRINT: PRINT TAB(3); "SET PRINTER TO CONDENSED MODE & TYPE THE MESSAGE IN HERE."
800 PRINT@985, "< HIT ENTER >";: PRINT@588, CHR$(94);
```

Initialize variables.

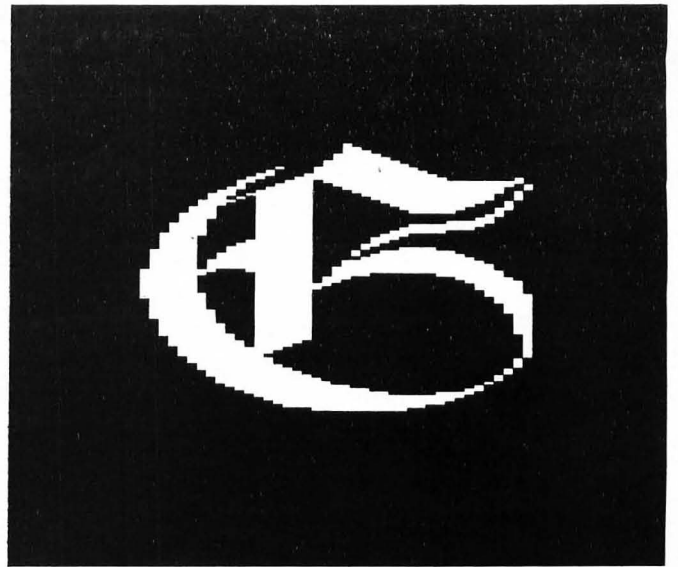
```
850 X=589: MS$="": DT$(50)=CHR$(34)
```

Get input from operator.

```
900 PRINT@X, CHR$(93);: L$=INKEY$: IFL$="" THEN 900 ELSE IFL$=CHR$(13) THEN 1100
950 IFL$=CHR$(8) AND LEN(MS$)>0 THEN PRINT@X, " ";: X=X-1: L$="": MS$=LEFT$(MS$, LEN(MS$)-1): GOTO 900 ELSE IFL$=CHR$(8) THEN 900
1000 IFL$=CHR$(9) OR L$=CHR$(10) OR L$=CHR$(91) THEN 900
1050 PRINT@X, L$: X=X+1: PRINT@X, CHR$(93);: MS$=MS$+L$: GOTO 900
1100 IF MS$="" THEN PRINT: PRINT TAB(16); "< NO MESSAGE, PLEASE TRY AGAIN >": GOTO 475
1125 PRINT@985, "      ";
```

Check for legal operator input.

```
1150 FOR I=1 TO LEN(MS$): L$=MID$(MS$, I, 1)
1200 FOR J=1 TO NLT
1250 IF D$(J)=L$ THEN IFL$=CHR$(34) THEN IF QUOTE=0 THEN PTR(I)=50: QUOTE=1: GOTO 1400 ELSE PTR(I)=51: QUOTE=0: GOTO 1400 ELSE PTR(I)=J: GOTO 1400
1300 NEXT J
1350 PRINT@41, "< MESSAGE HAS AN UNRECOGNIZABLE CHARACTER "; L$;
```



```
, PLEASE RETYPE >";: GOTO 475
1400 NEXT I
```

Display heading.

```
1450 CLS: PRINT: PRINT TAB(11); "G O T H I C   L E T T E R   P R I N T E R"
1500 PRINT TAB(11); " * * * * *   * * * * *   * * * * * * * * * * "
1550 PRINT@345, "YOUR MESSAGE": PRINT@409, "-----": PRINT: PRINT TAB(ABS((31-(LEN(MS$)/2)))) MS$
```

Main program loop. Get one character from operator's message.

```
1600 FOR I=1 TO LEN(MS$): S%=MID$(MS$, I, 1)
1650 IF S%="" THEN FOR X=1 TO 15: LPRINT: NEXT X: NEXT I: GOTO 1800
1700 GOSUB 2000
1750 NEXT I
```

Offer manual paper advance or restart of program.

```
1800 PRINT@911, STRING$(34, " ");: PRINT@787, "< HIT 'ENTER' TO RESTART >";
1850 PRINT@848, "< DEPRESS 'CHR$(91)' TO ADVANCE PAPER >";
1900 FOR I=1 TO 10000: K$=INKEY$: IF K$=CHR$(13) THEN RESTORE: GOTO 475 ELSE IF PEEK(15168)=8 THEN LPRINT
1950 NEXT I: CLS: PRINT "TYPE 'RUN' TO RESTART PROGRAM": END
```

Display gothic character which is being printed.

```
2000 PRINT@913, "< CREATING THE CHARACTER 'S%' >";
```

Read data for one gothic letter.

```
2010 FOR I=1 TO NLT-1: READ A$: NEXT I
2025 FOR J=1 TO PTR(I)-1
2150 READ C
2200 FOR I%=1 TO C: READ A$(I%): NEXT I%
2250 NEXT J: RESTORE
```

Create one print line of character data.

```
2300 FOR J=1 TO C: ST$=""
```


3950 DATAB27C10B10C50,B26C10B26C2B31C4,B26C9B28C3B28C7,B26C8B30C5B24C10,B26C8B31C6B21C14,B26C7B33C5B1C2B17C14,B26C7B33C6B2C2B14C14,B26C6B35C6B2C3B11C14,B26C6B35C7B2C3B9C14,B26C5B36C7B3C3B7C14,B26C5B36C8B3C3B5C14,B27C4B36C8B4C3B3C14

4000 DATAB27C4B36C8B4C3B3C13,B28C3B36C8B4C3B3C12,B28C4B34C8B5C3B3C11,B29C3B34C8B6C3B2C10,B30C2B33C8B7C3B3C9,B31C2B31C8B8C3B4C8,B32C2B28C9B10C3B4C7,B34C1B25C10B12C3B4C6,B35C2B21C10B16C3B4C4,B37C2B16C11B20C3B4C2,B39C3B7C16B24C3B2C2

4050 DATAB42C19B31C2

4100 DATA38,B26C3B58C4,B28C4B57C6,B29C5B58C6,B29C7B23C1B33C7,B29C8B23C3B31C7,B30C8B23C5B29C7,B30C8B24C6B27C8,B30C9B23C7B26C8,B30C9B23C8B25C8,B29C9B24C9B23C8,B29C9B24C10B22C8,B29C8B24C11B22C8,B29C65B1C8,B28C8B1C5B81C7,B27C9B2C5B81C7

4150 DATAB27C8B5C57B2C5,B1C1B24C8B8C57B3C3,B2C2B22C8B12C35B13C12,B3C3B20C8B42C2B15C12,B4C3B19C8B43C2B14C11,B4C5B17C8B43C3B13C10,B5C5B16C8B42C5B12C10,B6C5B16C8B40C7B11C10,B7C6B15C7B39C9B11C8,B8C7B15C6B38C10B10C8,B9C8B16C4B36C12B10C7

4200 DATAB10C9B17C3B34C11B11C7,B11C11B51C10B13C6,B13C12B47C11B13C6,B14C6B815C6,B16C66B16C5,B19C62B18C4,B21C60B19C4,B23C58B21C3,B27C54B23C2,B76C6,B78C5,B81C3

4250 DATA17,B26C1B60C4,B26C3B60C6,B26B1C4B61C6,B26B2C4B27C1B33C7,B26B3C4B27C3B31C7,B26B4C5B26C5B29C7,B26B4C6B26C6B27C8,B26B4C7B25C7B26C8,B26B4C8B24C8B25C8,B26B4C9B23C9B23C8,B26B3C11B22C10B22C8,B26B3C11B21C11B22C8,B26B2C66B1C8,B26B1C68B1C7

4300 DATAB26B1C69B1C7,B26C71B2C5,B26B2C71B3C3

4350 DATA22,B35C13,B31C20,B29C24,B28C27B32C4,B28C15B3C10B33C6,B27C12B12C6B35C6,B27C9B18C5B8C1B25C7,B26C8B22C5B4C2B27C7,B26C7B25C8B29C7,B26C6B29C4B30C8,B27C4B64C8,B27C4B64C8,B28C4B46C8B8C8,B29C4B36C21B4C8,B30C4B30C29B1C8,B31C5B20C3B81C8

4400 DATAB33C62B1C7,B35C61B1C7,B37C36B16C8B2C5,B40C26B26C7B3C3,B45C14B36C6B3C2,B98C5

4450 DATA41,B26C3B58C4,B28C4B57C6,B29C5B58C6,B29C7B23C1B33C7,B29C8B23C3B31C7,B30C8B23C5B29C7,B30C8B24C6B27C8,B30C9B23C7B26C8,B30C9B23C8B25C8,B29C9B24C9B23C8,B29C9B24C10B22C8,B29C8B24C11B22C8,B29C65B1C8,B28C8B1C5B81C7,B27C9B2C5B81C7

4500 DATAB27C8B5C57B2C5,B26C8B8C57B3C3,B26C8B12C56B2C2,B26C8B25C2B16C2B15C12,B26C8B26C2B16C2B13C12,B26C8B26C3B16C2B12C11,B26C8B26C4B14C4B11C10,B26C8B26C5B11C7B10C10,B27C8B25C6B8C10B9C10,B29C7B24C8B5C13B9C8,B30C6B22C10B2C16B8C8

4550 DATAB33C4B19C10B11C7,B36C3B14C12B3C14B13C7,B45C20B4C11B16C6,B36C28B6C8B18C6,B32C31B8C5B21C6,B30C32B10C2B24C5,B29C30B40C4,B28C23B49C4,B27C16B59C3,B27C11B66C2,B26C9,B26C7,B26C6,B27C4,B28C3

4600 DATA37,B26C1B60C4,B26C3B60C6,B27C4B61C6,B28C4B27C1B33C7,B29C4B27C3B31C7,B29C6B26C5B29C7,B30C6B26C6B27C8,B31C6B25C7B26C8,B31C7B24C8B25C8,B32C7B23C9B23C8,B32C8B22C10B22C8,B32C8B21C11B22C8,B32C62B1C8,B31C9B1C54B1C7,B31C10B1C54B1C7

4650 DATAB30C11B3C53B2C5,B30C11B5C53B3C3,B29C11B9C53B2C2,B29C11B

54C12,B28C12B53C10,B28C11B54C9,B27C12B55C7,B27C11B57C5,B27C10B59C4,B26C10B61C3,B26C10B62C3,B26C9B65C2,B26C9,B27C8,B28C7,B28C7,B29C6,B30C5,B32C3,B34C2,B36C2

4700 DATA55,B26C2B59C4,B28C3B58C6,B29C4B59C6,B30C4B25C1B33C7,B30C6B24C3B31C7,B30C7B24C5B29C7,B30C8B24C6B27C8,B30C9B23C7B26C8,B30C9B23C8B25C8,B30C10B22C9B23C8,B30C10B22C10B22C8,B29C11B21C11B22C8,B28C66B1C8,B26C69B1C7,B27C69B1C7,B29C68B2C5

4750 DATAB31C68B3C3,B35C67B2C2,B94C12,B93C12,B59C1B32C12,B60C3B28C12,B61C5B24C12,B62C6B21C12,B62C7B19C12,B62C8B17C12,B31C1B30C9B15C12,B30C3B29C10B13C12,B29C5B27C11B12C12,B28C67,B27C67,B26C67,B27C65,B28C63,B29C61,B30C5B25C10B19C3

4800 DATAB31C3B26C9B22C3,B32C1B28C8B24C3,B62C7B26C3,B63C6B27C4,B64C5B26C7,B65C5B25C9,B68C3B23C12,B31C1B39C1B22C11,B30C3B60C11,B29C5B59C10,B28C74,B27C75,B26C76,B27C75,B28C75,B29C74,B30C5B64C5,B31C3B66C4,B32C1B69C3

4850 DATA40,B26C2B34C3B31C2,B27C3B34C4B29C3,B28C4B33C5B28C4,B29C5B32C6B26C5,B29C6B31C7B25C6,B30C6B30C8B23C8,B30C7B29C10B21C10,B30C76,B30C76,B30C75,B30C10B47C17,B29C11B43C19,B29C12B39C20,B28C13B36C20,B26C15B33C20,B28C13B30C20,B30C11B27C20

4900 DATAB32C9B24C20,B33C8B21C20,B36C4B19C20,B38C1B17C20B17C1,B53C20B19C4,B50C20B21C7,B47C20B24C9,B44C20B27C11,B41C20B30C13,B38C20B33C15,B35C20B36C13,B32C20B39C12,B29C20B43C11,B28C18B46C10,B27C75,B26C76,B26C76,B26C11B58C7,B26C10B60C6

4950 DATAB27C8B62C6,B29C6B63C5,B31C4B65C4,B33C3B66C3

5000 DATA38,B59C9,B54C21,B48C31,B45C38,B42C43,B40C47,B38C23B6C2B9C11,B36C20B12C3B10C2B1C7,B34C18B16C5B9C2B4C5,B33C16B19C6B9C2B5C4,B32C15B21C7B9C2B6C3,B31C14B22C9B9C2B7C2,B30C58,B29C12B3C45,B28C12B5C45,B28C11B7C46,B27C11B10C45,B27C10B15C43

5050 DATAB26C10B58C3,B27C8B58C6,B29C5B58C10,B30C3B58C15,B31C2B58C14,B32C2B56C14,B34C2B53C15,B35C2B51C15,B37C2B47C16,B38C3B44C16,B40C4B40C16,B41C6B35C17,B43C7B29C18,B45C8B23C20,B47C11B14C22,B49C41,B52C36,B54C31,B57C25,B60C19

5100 DATA38,B27C3B58C4,B28C4B57C6,B29C5B58C6,B29C7B23C1B33C7,B29C8B23C3B31C7,B30C8B23C5B29C7,B30C8B24C6B27C8,B30C9B23C7B26C8,B30C9B23C8B25C8,B29C9B24C9B23C8,B29C9B24C10B22C8,B29C8B24C11B22C8,B29C65B1C8,B28C8B1C5B81C7,B27C9B2C5B81C7

5150 DATAB27C8B5C57B2C5,B26C8B8C57B3C3,B26C8B12C56B2C2,B26C8B12C4B25C12,B26C8B30C7B23C11,B26C8B29C10B20C11,B26C8B28C13B18C11,B26C8B28C11B19C11,B27C8B26C10B21C11,B28C7B25C9B22C11,B30C6B24C8B23C11,B33C4B23C7B23C11,B36C3B20C7B24C10,B59C6B24C10

5200 DATAB59C5B24C10,B59C5B23C10,B60C3B23C10,B60C3B22C9,B60C3B20C10,B61C2B18C10,B62C2B15C10,B63C2B12C9,B65C18

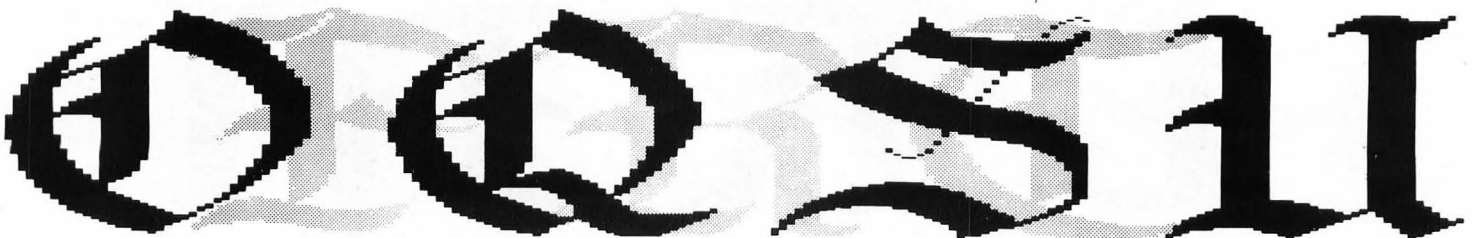
5250 DATA41,B59C9,B54C21,B48C31,B45C38,B42C43,B40C47,B38C23B6C2B9C11,B36C20B12C3B10C2B1C7,B34C18B16C5B9C2B4C5,B33C16B19C6B9C2B5C4,B32C15B21C7B9C2B6C3,B31C14B22C9B9C2B7C2,B30C50,B29C12B1C47,B28C12B3C47,B28C11B4C49,B27C11B5C4B1C45

5300 DATAB27C10B6C5B4C3,B26C10B7C6B45C3,B27C8B7C7B44C6,B29C5B7C9B42C11,B30C3B6C11B41C16,B31C2B4C12B42C15,B32C16B42C15,B31C15B43



C15, B29C16B43C15, B28C15B43C16, B27C14B44C16, B26C13B1C4B40C17, B26C11B4C6B35C17, B26C9B8C7B29C18, B26C8B11C8B23C19
 5350 DATAB26C8B13C11B14C21, B26C8B15C41, B27C7B18C36, B28C6B20C31, B29C5B23C25, B30C4B26C19, B31C4B30C9, B32C3, B34C2
 5400 DATA1, B26C3B58C4, B28C4B57C6, B29C5B58C6, B29C7B23C1B33C6, B29C8B23C3B31C7, B30C8B23C5B29C7, B30C8B24C6B27C8, B30C9B23C7B26C8, B30C9B23C8B25C8, B29C9B24C9B23C8, B29C9B24C10B22C8, B29C8B24C11B22C8, B29C6B1C8, B28C8B1C5B8C7, B27C9B2C5B8C7
 5450 DATAB27C8B5C57B2C5, B26C8B8C57B3C3, B26C8B12C56B2C2, B26C8B29C2B29C12, B26C8B29C2B29C11, B26C8B29C2B29C10, B26C8B30C2B28C10, B26C8B30C2B28C10, B27C8B29C2B28C9, B28C7B29C3B26C10, B30C6B28C4B25C10, B33C4B26C6B24C9, B36C3B23C8B22C10, B60C11B20C10
 5500 DATAB58C14B19C10, B30C43B18C10, B28C43B1C2B16C10, B27C44B2C2B14C10, B26C43B5C3B10C12, B26C41B9C4B5C14, B26C37B14C20, B26C5B48C16, B26C4B51C12, B26C4B54C6, B27C3, B28C3
 5550 DATA39, B26C3, B28C4, B30C4, B32C4, B33C5, B34C5B42C2, B35C5B39C5, B35C6B36C8, B36C6B33C11, B36C7B31C13, B37C7B29C15, B37C7B28C17, B37C8B26C19, B37C8B25C17B2C3, B37C9B10C2B11C16B6C3, B36C10B9C1B12C16B9C3, B36C10B8C1B12C16B10C5, B35C11B8C1B12C15B10C9
 5600 DATAB34C12B9C1B10C15B10C14, B33C12B11C2B7C15B10C13, B32C13B14C2B4C14B11C11, B31C13B18C16B11C11, B29C14B21C13B12C9, B26C17B20C13B12C10, B27C14B22C12B12C11, B27C13B22C12B1C2B10C11, B28C11B23C12B4C2B6C12, B29C8B24C12B8C2B3C11, B30C6B25C12B11C13
 5650 DATAB30C5B25C12B14C11, B31C2B27C12B14C11, B32C2B25C12B16C10, B33C2B23C12B17C10, B34C2B21C13B18C9B1C2, B35C3B18C13B20C8B4C2, B36C3B15C13B23C7B6C1, B37C4B10C14B27C6B6C1, B39C25B31C5B5C1, B41C20B36C4B3C1
 5700 DATA38, B59C9B23C1, B53C21B18C2, B48C31B15C2, B45C37B13C3, B42C43B11C4, B40C47B10C4, B38C23B17C11B9C4, B36C20B27C8B8C4, B34C18B14C2B19C5B7C5, B33C16B18C3B20C3B6C6, B32C15B21C4B20C3B4C6, B31C14B23C5B21C2B3C7, B30C13B25C6B21C2B2C7, B29C13B26C7B21C10
 5750 DATAB28C12B27C8B22C9, B27C60B9C10, B27C11B1C49B7C10, B27C10B3C49B5C10, B26C10B5C49B3C11, B26C9B7C49B1C11, B26C8B9C60, B26C8B57C11, B26C7B58C11, B26C7B57C11, B26C6B58C11, B26C6B58C11, B26C5B60C9, B26C5B60C9, B26C5B60C9, B27C4B61C8, B27C4B61C8
 5800 DATAB28C3B62C8, B28C4B62C7, B29C3B62C7, B30C3B62C6, B31C2B63C6, B32C2B63C5, B33C2B64C4
 5850 DATA37, B26C2B36C3B29C2, B27C4B34C5B28C3, B28C5B33C6B27C4, B29C5B33C6B26C5, B29C6B32C7B25C6, B29C7B31C8B24C7, B30C7B29C10B22C8, B30C76, B30C76, B30C75, B30C74, B29C73, B28C71, B28C12, B27C13, B26C14, B27C12, B28C11, B29C9, B30C7, B31C6B61C2, B32C4B63C4
 5900 DATAB33C2B65C4, B34C2B64C5, B35C2B62C7, B36C70, B34C72, B32C74, B30C76, B28C77, B26C78, B27C13B57C7, B28C10B60C6, B29C7B63C5, B30C4B67C4, B31C2B71C2, B32C2
 5950 DATA35, B26C2B36C3B31C2, B27C4B34C5B29C4, B28C5B33C6B28C4, B29C5B32C7B27C5, B29C6B30C9B26C6, B29C77, B30C76, B30C76, B30C76, B30C75, B30C74, B29C10B58C7, B28C11B59C6, B28C12B59C5, B27C13B61C4, B26C14B64C2, B27C12, B28C11, B29C9, B30C7, B31C6, B32C4, B33C2

6000 DATAB34C2, B35C2B51C9, B36C3B34C2B, B37C65, B39C64, B41C63, B43C62, B46C33B18C8, B49C14B36C7, B26B75C5, B26B76C3, B26B77C2
 6050 DATA48, B26C2B36C3B29C2, B27C4B34C5B28C3, B28C5B33C6B27C4, B29C5B33C6B26C5, B29C6B32C7B25C6, B29C7B31C8B24C7, B30C7B29C10B22C8, B30C76, B30C76, B30C75, B30C74, B29C73, B28C71, B28C13, B27C14, B26C15, B27C12B25C3, B28C11B26C5, B29C9B28C6B26C2
 6100 DATAB30C7B30C6B26C4, B31C6B30C7B26C4, B32C4B31C8B25C5, B33C3B30C10B23C7, B34C72, B32C74, B30C76, B28C78, B26C80, B26C79, B27C8B28C10B23C8, B28C5B31C8B25C7, B29C2B34C7B26C6, B30C2B34C6B27C5, B31C2B34C6B28C4, B32C2B35C5B30C2, B33C3B36C3, B34C4B49C6
 6150 DATAB35C7B41C14, B36C11B22C31, B37C65, B39C64, B41C63, B43C62, B46C33B18C8, B49C14B36C7, B53C6B42C5, B26B76C3, B26B77C2
 6200 DATA36, B26C3B64C4, B27C5B63C5, B28C6B63C6, B28C7B62C7, B29C7B18C3B40C8, B29C8B18C5B37C9, B28C9B19C6B34C10, B28C9B20C7B31C11, B28C10B20C8B27C13, B27C11B21C9B22C15, B27C11B21C10B18C17, B26C12B21C12B12C19, B32C6B22C12B7C21, B37C6B17C12B3C23
 6250 DATAB43C6B11C36, B49C6B4C33, B55C33, B59C24, B56C23, B52C23, B48C29, B44C29B4C6, B40C32B11C6, B36C36B17C6, B31C24B6C12B21C6, B29C21B13C10B21C12, B28C17B19C9B21C11, B27C15B24C8B20C11, B26C13B29C7B19C10, B26C11B33C6B19C9, B26C10B36C5B18C9
 6300 DATAB26C9B40C3B17C8, B27C8B62C7, B28C7B63C7, B29C6B64C6, B32C5B64C5
 6350 DATA36, B26C1B33C3B33C2, B26C3B33C4B31C4, B27C4B32C5B30C5, B28C4B31C7B28C6, B1C2B25C6B29C8B27C7, B3C3B22C7B28C9B25C9, B4C4B21C7B26C11B23C10, B5C5B19C77, B6C5B18C77, B6C6B17C77, B7C6B16C76, B7C6B16C75, B7C7B14C73, B7C7B14C11B52C3, B7C8B12C12B54C3
 6400 DATAB7C8B11C12B57C3, B7C9B12C10B58C4, B6C10B14C8B57C7, B6C10B14C8B58C9, B6C10B18C3B57C12, B5C11B20C3B55C11, B5C11B22C3B53C10, B4C11B25C3B50C11, B4C11B27C3B48C10, B3C12B29C3B46C10, B2C12B20C69, B1C13B16C72, B2C11B15C74, B3C9B14C76, B4C7B13C78
 6450 DATAB5C5B12C80, B6C3B11C16B60C7, B7C3B8C16B63C6, B9C23B66C6, B1C18B71C5, B15C11B77C3
 6500 DATA39, B26C3B52C5, B28C3B46C14, B29C4B42C19, B30C5B39C5B6C11, B30C6B37C3B11C11, B31C7B35C2B13C12, B31C10B32C1B15C13, B31C11B47C15, B32C8B2C2B45C17, B32C9B3C2B43C16, B32C9B5C2B40C16, B32C10B6C2B11C5B22C15, B32C10B8C2B11C7B17C15
 6550 DATAB32C11B9C2B10C8B15C14, B32C11B11C2B8C9B13C14, B32C11B13C2B6C9B13C14, B32C11B15C2B4C9B13C14, B32C12B16C2B2C9B13C13, B32C12B18C11B12C14, B32C12B20C9B12C14, B32C12B20C9B12C14, B32C12B20C9B12C14, B31C13B20C9B13C13, B31C13B20C10B12C13
 6600 DATAB31C13B20C9B1C2B10C13, B30C14B20C9B3C2B9C12, B29C15B20C9B5C2B7C12, B29C15B20C9B7C2B5C12, B28C15B22C8B9C2B4C11, B27C16B24C7B10C2B2C11, B26C17B28C5B10C2B1C10, B27C15B46C12, B28C14B12C2B34C10, B30C12B13C2B35C9, B32C11B11C4B36C7, B34C10B9C5B38C6
 6650 DATAB36C10B5C6B41C4, B38C18B44C3, B40C14B48C2
 6700 DATA10, B30C1B59C1, B31C2B57C1, B32C2B56C2, B33C60, B34C60, B35C5B, B36C56, B37C54, B38C2, B39C2
 6750 DATA19, B30C7B34C15, B31C6B1C4B30C15, B31C7B4C4B27C15, B32C7B7C



DATA DATA DATA DATABASE DATABASE DATABASE

by Mark Pelczarski
TRS-80® translation by
Robert Jacobs

Database is a set of file manager programs designed to run on a TRS-80® Model I/III with 48K and one disk drive. Operation in 32K is possible with modification of the CLEAR statement at the beginning of the programs.

TRS-80® SEQUENTIAL FILE DATABASE

The TRS-80® version of *SoftSide's Database* appears in its entirety starting on the next page. It follows the Apple version published in October, 1981, very closely, and the instructions for using the Apple version are appropriate for the TRS-80®. There are a few differences of style and a few minor modifications which spring from the difference between the operating systems of the Apple and TRS-80®. These will be commented upon as they arise in the description below.

The user of this database is first asked whether he or she wishes to begin a new file or load an old one. The files established by the database carry their own headings and format; if an existing file is to be loaded, no further decisions are required of the operator, who is presented with the main menu after the data is loaded. If a new file is to

be created, the number of headings and the name of each must be entered.

The main menu offers most of the usual database manipulations. One may add, delete, alter, sort or print data. The search routine is called if one chooses to print, delete or alter records. As in the original version it is possible to operate on one item, a subset of items, or all items. The heading under which one wishes to select is chosen from the search menu; then a criterion is elected (greater than, less than, or equal to), and a value for comparison is entered.

A partial value may be used by substituting an asterisk for the remainder of the record. To use Mark Pelczarski's example, all zip codes beginning "60" could be selected by entering "60*" as the comparison value. The program loops back to the search routine with each selection, so that one may either add criteria or begin the search. If necessary, it is also possible to return to the main menu and start over at this point.

One difference between the Apple and TRS-80® versions of the database is that the latter telescopes the print options into one menu which offers the choice of printing either to the display or to the lineprinter with either a default or custom format. One keystroke suf-

fices. This menu also warns if the lineprinter is not ready.

On selection of one of the hard copy options, BASIC's TIMES\$ is printed. This feature can be disabled without difficulty if desired. If one of the default format options is chosen the search routine already discussed is called; otherwise the user must either load or construct a custom format.

Construction of the format is unchanged from the earlier versions. In making the format (don't forget to save it!) one chooses among the (1) Heading, (2) Item (this is the actual data item), (3) Tab number, (4) Next line, (5) String, and (6) End.

Where necessary the program prompts for the appropriate value. Thus if "tab" is selected, the tab number (not the number of blank spaces!) has to be entered. If "string" is selected, the text of the string must be supplied. If "1" is supplied after the entry of "4" (Next line), the printout drops down one line, and so on. The ability to prepare reports as one wishes is an important feature for a database program, and this one is quite versatile.

Users of disk operating systems which permit access to directories from BASIC may wish to enable lines 270, 380 and 600 by removing the apostrophes. This will allow a look at the available data files. NEWDOS 2.1 requires that there be several thousand bytes free for this technique to work; one would have to reduce the string space cleared in line 100 to 17,000 (for a 48K machine) in order to use this command. In NEWDOS-80 Versions 1 and 2, the string space may remain high. Note that in line 600 the drive number is chosen. This may be altered for your own purposes; indeed, some of the better operating systems allow a format like `CMD"DIR 2 /DAT"` which displays only files with the extension /DAT, a great convenience.

Display of remaining free space has been left optional in the main menu. This is because the `FRE(A$)` function, which forces garbage collection, is very slow with large data files.

Many enhancements for this database are possible, some of them

branching out into functions not accessible to the TRSDOS user. Among the obvious possibilities are the use of Machine Language sort and search routines to speed the program, as well as the random access and numerical data enhancements suggested by Mark Pelczarski in his October article. It is to be hoped that we will see a number of these soon.

Variables

A\$: User answers.
 AS: Accept switch for search routine.
 B\$: Formatted output string.
 BS: Condition switch.
 C\$(n): Test conditions for search.
 C1(n), C2(n): Store less, equal or greater choices for search routine.
 FS: Filename.
 FS(n): Character string holding format codes.
 FS: Format switch.
 H\$: Heading string.
 I, I1, I2, I9: Loop and array counters.
 II, J, J1, J2: Loop and array counters.
 MX: Maximum number of records.
 N: Loop counter, format routine.
 NF: Number of format instructions.
 NH: Number of headings.
 NI: Number of items.
 P1, P2: Poke addresses for directing output to printer.
 PR: Printer driver address.
 RS: Return switch; when equal to 1, control returns to menu.
 S1, S2: Poke addresses for directing output to display.
 SB: Directs search subroutine to change, delete or print routine.
 SC: Screen driver address.
 SS: Save switch.
 T: Format array counter (3800-3960). At 7100-7160 used as temporary holding variable in sort routine.
 TS: Temporary string; holds decision to save file at 500 and is used in sort routine for temporary storage.

```

#####
$ TRS-80 Mod 1/3 DISK BASIC $
$ "SEQUENTIAL DATABASE" $
$ Author: Mark Pelczarski $
$ Translator: Robert Jacobs $
$ (c) 1982, SoftSide $
#####
  
```

Main Control routine.

```

100 CLS: CLEAR20000: MX=150
102 SC=16414: S1=PEEK(SC): S2=PEEK(SC+1)
103 PR=16422: P1=PEEK(PR): P2=PEEK(PR+1)
105 DIM C$(7), C1(7), C2(7), F$(5)
110 PRINT (I) INITIALIZE A NEW DATA SET"
120 PRINT (L) LOAD A PREVIOUSLY SAVED DATA SET ";
130 GOSUB60000
140 IFA$="L"ORA$="1" GOSUB1000: GOTO200
150 IFA$="I"ORA$="i" GOSUB1500: GOTO200
160 GOTO100
200 CLS: PRINT (S) Save current data"
  
```

```

210 PRINT (P) Print data"
220 PRINT (A) Add data"
230 PRINT (C) Change a record"
240 PRINT (D) Delete a record"
260 PRINT (T) Sort"
270 PRINT (N) New data file"
280 'PRINT (F) File names"
290 PRINT (Q) Quit"
295 'PRINT (E) Examine free space"
300 GOSUB 60000: PRINT
310 IFA$="S"ORA$="s" GOSUB2000: GOTO200
320 IFA$="P"ORA$="p" GOSUB3000: GOTO200
330 IFA$="A"ORA$="a" GOSUB4000: GOTO200
350 IFA$="C"ORA$="c" THEN SB=3: GOSUB8000: GOTO200
360 IFA$="D"ORA$="d" THEN SB=4: FS=1: GOSUB8000: GOTO200
370 IF A$="T"ORA$="t" THEN GOSUB7000: GOTO200
380 'IF A$="F"ORA$="f" THEN GOSUB600: GOTO200
390 IFA$="Q"ORA$="q"ORA$="N"ORA$="n" THEN 500
395 IFA$="E"ORA$="e" THEN GOSUB699: GOTO200
397 IFA$<>"Q" AND A$<>"e" GOTO200
500 T=A$: IF SS=1 THEN 540
510 PRINT "CURRENT FILE IS NOT SAVED.": PRINT "CANCEL COMMAND (Y/N)
? "; GOSUB60000
520 IFA$="Y"ORA$="y" THEN 200
530 IFA$<>"N" AND A$<>"n" THEN 510
540 IFT$="N" OR T$="n" THEN 100
550 END
600 'CMD"DIR": GOSUB60000: RETURN
699 PRINT "YOU HAVE SPACE FOR"; FRE(A$); "CHARACTERS": GOSUB60000: RE
TURN
  
```

Load subroutine.

```

1000 LINE INPUT "FILENAME: "; F$
1005 ON ERROR GOTO1310
1010 OPEN "I", 1, F$
1020 INPUT #1, NH, NI
1120 DIM H$(NH), I$(MX, NH)
1140 FOR I=0 TO NH: INPUT #1, H$(I): NEXT
1200 IF NI=-1 THEN 1280
1240 FOR I=0 TO NI
1250 FOR J=0 TO NH
1260 INPUT #1, I$(I, J)
1270 NEXT J
1280 NEXT I
1290 CLOSE
1300 ON ERROR GOTO0: SS=1: RETURN
1310 PRINT "FILE NOT FOUND": GOSUB60000: RESUME100
  
```

Initialize subroutine.

```

1500 LINE INPUT "Give your file a name ==> "; F$
1510 IF F$="" THEN 1500
1520 INPUT "How many headings"; NH
1530 IF NH<1 THEN 1520
1540 NH=NH-1: NI=-1
1560 DIM H$(NH), I$(MX, NH)
1570 FOR I=0 TO NH
1580 PRINT "Heading #"; I+1; ": INPUT " : "; H$(I)
1590 NEXT I
1600 SS=0: RETURN
  
```

Write subroutine.

```

2000 PRINT"Use ";F$;" (Y/N)?";:GOSUB60000
2010 IFA$="Y"ORA$="y"THEN2050
2020 IFA$<>"N"ANDA$<>"n"THEN2000
2030 LINEINPUT"NAME: ";F$
2040 IFF$=""THEN2030
2045 ON ERROR GOTO 2290
2050 OPEN"D",1,F$
2060 PRINT#1,NH;NI
2070 FORI=0TONH
2080 PRINT#1,H$(I)
2090 NEXT
2220 IFNI=-1THEN2270
2230 FORI=0TONI
2240 FORJ=0TDNH
2250 PRINT#1,I$(I,J)
2260 NEXT:NEXT
2270 CLOSE
2280 ON ERROR GOTO0:SS=1:RETURN
2290 PRINT"DISK ERROR":GOSUB60000:RESUME200

```

Print subroutine.

```

3000 IF NI=-1THENGOSUB9000:RETURN
3005 CLS:PRINT"(A) SCREEN PRINT, DEFAULT FORMAT
(B) SCREEN PRINT, SELECT FORMAT
(C) LINE PRINT, DEFAULT FORMAT
(D) LINE PRINT, SELECT FORMAT":IFPEEK(14312)<>63ANDPEEK(293)<>73
THENPRINT:PRINT"PRINTER NOT READY!":GOSUB60000ELSEGOSUB60000
3006 IFA$="A"ORA$="a"THENFS=1:GOTO3040
3007 IFA$="B"ORA$="b"THENGOSUB10000:FS=2:GOTO3040
3008 IFA$="C"ORA$="c"THENFS=1:SB=2:LPRINTCHR$(138):LPRINTTIME$:G
OTO3060
3009 IFA$="D"ORA$="d"THENGOSUB10000:FS=2:SB=2:LPRINTCHR$(138):LP
RINTTIME$:GOTO3060
3010 IF A$<>"D"ANDA$<>"d"THEN3005
3040 SB=1:CLS:PRINT"AFTER EACH RECORD <M> WILL RETURN YOU TO MEN
U":PRINT
3050 PRINT"Press any key . . .":GOSUB60000
3060 GOSUBB010
3090 IFSB=2THENPOKESC,S1:POKESC+1,S2
3100 RETURN

```

Print one record to the screen.

```

3300 ON FSGOSUB 3700,3800
3340 GOSUB60000:IFA$="M"ORA$="m"THENRS=1
3350 RETURN
3600 ON FS GOSUB 3700,3800:RETURN

```

Print one record default format.

```

3700 CLS:PRINT" ":PRINT"RECORD ";I+1:PRINT" "
3710 FORJ=0TDNH
3720 PRINTH$(J),I$(I,J)
3730 NEXTJ
3740 RETURN

```

Print one record custom format.

```

3800 CLS:J=1:T=0:B$=""

```

```

3820 J1=VAL(MID$(F$(T),J,1)):J=J+1
3830 IFJ1<5THENN=VAL(MID$(F$(T),J,2)):J=J+2
3840 ON J1 GOTO3850,3860,3870,3890,3910,3970
3850 A$=H$(N):GOTO3950
3860 A$=I$(I,N):GOTO3950
3870 B$=LEFT$(B$,N-1):IFLEN(B$)<N-1THENFORJ2=LEN(B$)TON-2:B$=B$+
" ":NEXT
3880 GOTO3960
3890 PRINTB$:IFN>1THENFORJ2=2TON:PRINT" ":NEXT
3900 B$="":GOTO3960
3910 IFJ>LEN(F$(T))THENT=T+1:J=1
3920 J2=J
3930 IFMID$(F$(T),J2,1)<>"!"THENJ2=J2+1:GOTO3930
3940 A$=MID$(F$(T),J,J2-J):J=J2+1
3950 B$=B$+A$
3960 IFJ>LEN(F$(T))THENT=T+1:J=1
3965 GOTO3820
3970 PRINTB$:RETURN

```

Add subroutine.

```

4000 SS=0:NI=NI+1
4005 PRINT:PRINT"Record";NI+1:PRINT
4010 FORJ=0TDNH
4020 PRINTH$(J);:INPUT" : ";I$(NI,J)
4030 NEXTJ
4040 RETURN

```

Change subroutine.

```

5000 PRINT:PRINT"(C) Change item, (K) Keep item, or (R) Keep rem
ainder of record"
5030 PRINT:PRINT"Record ";I+1
5040 CS=1:RS=0:FORJ=0TDNH
5050 PRINT:PRINT H$(J);" : ";I$(I,J);" ";
5055 IFRS=1THENPRINT:GOTO5090
5060 GOSUB 60000:IFA$<>"C"ANDA$<>"c"ANDA$<>"K"ANDA$<>"k"ANDA$<>"
R"ANDA$<>"r"THEN5060
5065 PRINTCHR$(ASC(A$)+32*(A$>"Z"))
5070 IFA$="K"ORA$="k"THEN5090
5075 IFA$="R"ORA$="r"THENRS=1:GOTO5090
5080 PRINT H$(J);:INPUT" : ";I$(I,J)
5085 CS=0
5090 NEXTJ
5095 RS=0
5100 IFCSS=0 THENSS=0
5110 RETURN

```

Delete subroutine.

```

6000 PRINT:PRINT"DELETE THIS RECORD? ";
6070 GOSUB60000:IFA$<>"Y"ANDA$<>"y"ANDA$<>"N"ANDA$<>"n"THEN6070
6080 PRINTCHR$(ASC(A$)+32*(A$>"Z")):IFA$="N"ORA$="n"THEN6150
6100 FOR I1 = I+1 TO NI
6110 FOR J=0TD NH
6120 I$(I1-1,J)=I$(I1,J)
6130 NEXTJ:NEXTI1
6135 FORJ=0TDNH:I$(NI,J)="" :NEXT
6140 NI=NI-1:SS=0:I=I-1
6150 RETURN

```

Sort subroutine.


```

7000 IFNI=-160SUB9000:RETURN
7010 CLS:FORJ=0T0NH
7020 PRINT@128+(J*32), "(" ; J+1 ; " ) " ; H$(J)
7030 NEXTJ:PRINT
7040 INPUT "SORT ON WHICH HEADING";J1
7045 J1=J1-1
7050 IFJ1<0ORJ1>NH:RETURN
7060 PRINT" (A) Ascending or (D) Descending":60SUB60000
7070 IFA$="A"ORA$="a"THENA=1:GOTO7100
7080 IFA$="D"ORA$="d"THENA=2:GOTO7100
7090 GOTO7060
7100 FORI=0TONI-1
7110 T=I
7120 FORI1=T+1TONI
7122 PRINT@960, I ; I1 ;
7125 ONAGOTO7130, 7140
7130 IFI$(I1, J1)<I$(T, J1)THENT=I1
7135 GOTO7145
7140 IFI$(I1, J1)>I$(T, J1)THENT=I1
7145 NEXTI1
7150 IFT=ITHEN7180
7155 FORJ=0T0NH
7160 T$=I$(T, J):I$(T, J)=I$(I, J):I$(I, J)=T$
7170 NEXTJ
7180 NEXTI
7200 SS=0:RETURN

```

Search subroutine.

```

8000 IFNI=-1THENG0SUB9000:RETURN
8010 I1=0: I2=NI: J=0: C1(0)=-1: BS=1
8015 CLS:PRINT"SEARCH CRITERIA:":PRINT
8020 PRINT" 0 ) RECORD NUMBER"
8030 FORI=0T0NH:PRINT@(((I+1)*32)+128), I+1 ; " ) " ; H$(I):NEXTI
8035 PRINT:PRINTNH+2 ; " ) BEGIN"
8036 PRINTNH+3 ; " ) RETURN TO MENU"
8040 PRINT@896, ; INPUT"WHICH FIELD";I:IFI<0ORI>NH+3THEN8040
8045 IFI=NH+2THENC1(J)=-1:GOTO8150
8046 IFI=NH+3 THEN RETURN
8050 C1(J)=I-1
8055 CLS:IFI=0PRINT@128, "SELECTING ON RECORD NUMBER":GOTO8060ELS
E:PRINT@128, "FIELD SELECTED: " ; H$(I-1)
8060 PRINT@256, "INDICATE (1) SMALLER, (2) EQUAL, OR (3) LARGER"
:60SUB60000:C2(J)=VAL(A$):IFC2(J)<10RC2(J)>3THEN8060
8080 PRINT@384, "COMPARED TO: " ; :IFC1(J)=-1THEN8100
8090 INPUT" " ; C$(J):J=J+1:IFJ>7THEN8160
8095 GOTO8015
8100 INPUT" " ; I:IFI<1ORI>NI+1THEN8100
8105 I=I-1
8110 IFC2(J)=1THENI2=I
8120 IFC2(J)=2THENI1=I:I2=I
8130 IFC2(J)=3THENI1=I
8140 GOTO8015
8150 IFJ<2THEN8200
8160 PRINT@896, "1) ITEM MUST MEET ALL CONDITIONS:PRINT"2) ITEM
MAY MEET ANY CONDITION";:60SUB60000:BS=VAL(A$):IFBS<10RBS>2THEN8
160
8200 PRINT"SEARCHING . . .":RS=0:IFSB=2THENPOKESC, P1:POKESC+1, P2
8250 I=I-1
8252 FORI9=I1TOI2:I=I+1
8255 AS=0:FORJ=0TO7

```

```

8260 IFC1(J)=-1THENJ=7:GOTO8345
8270 ONC2(J)GOTO8280, 8290, 8310
8280 IFI$(I, C1(J))<=C$(J)THEN8330
8285 GOTO8340
8290 IFI$(I, C1(J))=C$(J)THEN8330
8295 IFRIGHT$(C$(J), 1)<>"*"THEN8340
8298 T=LEN(C$(J))-1:IF LEN(I$(I, C1(J)))<TTHEN8340
8302 IFLEFT$(I$(I, C1(J)), T)=LEFT$(C$(J), T)THEN8330
8305 GOTO8340
8310 IFI$(I, C1(J))>=C$(J)THEN8330
8320 GOTO8340
8330 IFBS=2THENAS=1:J=7
8335 GOTO8345
8340 IFBS=1THENAS=2:J=7
8345 NEXTJ
8350 IFAS=0ANDBS=1THEN8355
8352 IFAS<>1THEN8380
8355 IFSB=1THENG0SUB3300
8360 IFSB=2ORSB=4THENG0SUB3600
8365 IFSB=3THENG0SUB5000
8370 IFSB=4THENG0SUB6000
8375 IFRS=1 THENI9=I2
8380 NEXTI9
8390 PRINT"PRINT"THAT'S ALL":FORI1=1TO500:NEXT
8400 RETURN

```

Error subroutine number one.

```

9000 PRINT"There's no data in memory."
9010 FORI=1TO1000:NEXT:RETURN

```

Print Formatting.

```

10000 IFF$(0)=" "THEN10040
10010 PRINT"SAME FORMAT?"::60SUB60000
10020 IFA$="Y"ORA$="y"THENRETURN
10030 IFA$<>"N"ANDA$<>"n"THEN10010
10040 PRINT"(L) LOAD FORMAT, OR (C) CREATE FORMAT"::60SUB60000
10050 IFA$="C"ORA$="c"THEN10200
10060 IFA$<>"L"ANDA$<>"l"THEN10040
10090 ON ERROR GOTO10170
10100 INPUT"FORMAT NAME: ";A$
10110 OPEN"I", 2, A$
10130 INPUT#2, NF
10140 FORJ=0T0NF:INPUT#2, F$(J):NEXT
10150 CLOSE2
10160 ON ERRORGOTO0:RETURN
10170 PRINT"FORMAT NOT FOUND":60SUB60000:RESUME200
10200 NF=0:J=0:F$(0)=" "
10210 CLS:PRINT"Start in the upper left corner and work across e
ach line"
10220 PRINT"1:HEADING, 2:ITEM, 3:TAB, 4:NEXT LINE, 5:STRING, 6:E
ND":INPUTJ1
10230 IFJ1<1ORJ1>6THEN10220
10240 F$(NF)=F$(NF)+RIGHT$(STR$(J1), LEN(STR$(J1))-1):J=J+1
10250 ONJ1GOTO10260, 10260, 10300, 10300, 10350, 10400
10260 FORT=0T0NF:PRINT@128+(T*32), T+1 ; " ) " ; H$(T):NEXT
10270 INPUT"WHICH";T:T=T-1:IFT<0ORT>NTHEN10270
10280 GOTO10310
10300 INPUT"HOW MANY";T:IFT<1ORT>99THENPRINT"OUT OF RANGE.":GOTO
10300

```

```

10310 A$=RIGHT$(STR$(T),LEN(STR$(T))-1):IFT<10THENA$="0"+A$
10320 F$(NF)=F$(NF)+A$:J=J+2
10330 GOTO10380
10350 INPUT"STRING: ";A$:A$=A$+" "
10360 IFLEN(A$)+J>255THENNF=NF+1:J=0:F$(NF)=""
10370 F$(NF)=F$(NF)+A$:J=J+LEN(A$)
10380 IFJ>252THENNF=NF+1:J=0:F$(NF)=""
10390 GOTO10210
10400 INPUT"FORMAT NAME: ";A$
10405 ON ERROR GOTO10460
10410 OPEN"O",2,A$
10430 PRINT#2,NF:FORJ=0TONF:PRINT#2,F$(J):NEXT
10440 CLOSE2
10450 ON ERROR GOTO0:RETURN
10460 PRINT"DISK ERROR":GOSUB60000:RESUME10400
60000 A$="":A$=INKEY$:IFA$=""THEN60000ELSEPRINT:RETURN
    
```

TRS-80® Random File Database

The TRS-80® version of *SoftSide's* random-access database program differs from the Apple version in several respects. Under TRSDOS, random access files have to be "fielded," and in consequence, additional program code has been added at line 20000 to permit the computer to field each 256-byte disk sector and to distinguish automatically between subrecords within each physical record. The user only needs to know how much space to allocate to each item or heading in the data set; the program then puts as many subrecords as will fit into each 256-byte sector.

This fielding process has a few consequences for the operator of the database program. The first of these is that the user's data set may not be larger than 256 characters total. This limitation exists in the Apple and ATARI® random-access versions also, though for different reasons.

Secondly, it means that the user must give a fair amount of thought to the organization of data. Even divisors of 256 waste the least space. Thus, if a logical record were to be 64 bytes long, four such records would be put into each disk sector and no space would be wasted at all. If the user chooses a logical record length of 129, 127 bytes will be wasted in each sector.

The logical record length is established when initializing a new file. It is the sum of the lengths entered for the headings, and is of course figured out before setting the file's initial parameters.

Because TRSDOS will not write a record "0" it has proved simplest to alter the program's structure so that records are numbered from 1. This change also allows easy selection of subrecords by the program logic, which fields each physical record as B\$(I,J), where I indexes the item and J is the subrecord identifying number, again beginning from 1 and running to FF, where FF is the total number of subrecords in the sector.

Many different types and sizes of data sets may be manipulated with this program without changing the internal logic and with little waste of disk space. Vastly greater amounts of data can be entered than can be managed in the sequential database system, in which all the data is held in memory.

To give an example, if the user had devoted an entire 40-track disk to the data file, and if each record were 64 bytes long, this program could manage 1600 records; in double-density and with all other factors constant, 2880 records can be put in. 80-track drives and multiple-head drives increase the capacity accordingly. It is also possible to span disk within the same file by using standard TRSDOS functions.

The remainder of the program is nearly identical to the Apple version. Record deletions are handled by writing a "link" number, LK, to the last empty record, and the formatting, data entry, and sorting features are the same. Note that lines 270, 380, and 600 have been disabled by REMark apostrophes. If your operating system permits access to your disk directory from these BASIC lines, then analogous commands for your system will permit you to examine your data files without exiting the program.

Variables

In addition to those listed for sequential:

B\$(I,J): Fielding array; I counts item numbers, J indexes subrecord numbers.

CH: Array counter to change index array held in memory.

D%: Length of dummy string.

DS: Index string switch; when 0, search routine calls new data heading into memory.

DU\$: Dummy string in fielding routine.

FF: Number of subrecords in each physical record.

LK: Link; holds number of record last deleted so empty spaces may be filled.

P%(n): Integer array; holds sorted order of records.

PR%: Number of the physical record.

R: Logical record number.

TIS(n): Holds items to be sorted or searched.

```

#####
$ TRS-80 Mod 1/3 DISK BASIC $
$ "RANDOM DATABASE" $
$ Author: Mark Pelczarski $
$ Translator: Robert Jacobs $
$ (c) 1982, SoftSide $
#####
    
```

Main Control routine.

```

100 CLS: CLEAR20000
102 SC=16414: S1=PEEK(SC): S2=PEEK(SC+1)
103 PR=16422: P1=PEEK(PR): P2=PEEK(PR+1)
105 DIMC$(7),C1$(7),C2$(7),F$(5):CH=0
107 DEFFNSB$(A$)=LEFT$(A$+" ",INSTR(A$+" ",")-1)
110 PRINT"(I) INITIALIZE A NEW DATA SET"
120 PRINT"(L) LOAD A PREVIOUSLY SAVED DATA SET";
130 GOSUB60000
140 IFA$="L"ORA$="1"GOSUB1000:GOTO200
150 IFA$="I"ORA$="i"GOSUB1500:GOTO200
160 GOTO100
200 CLS:PRINT"(S) Save current data"
210 PRINT"(P) Print data"
220 PRINT"(A) Add data"
    
```

SAVE / on Software for TRS-80

— APPARAT'S FLEXTXT/80 — PUT'S PEP IN THE "EPSON"

FLEXTXT/80 provides a SCRIPSIT™ path to EPSON power. Users will:

- * Print superscripts and subscripts anywhere in text.
- * Underline any text (including super and sub scripts).
- * Mix 10/inch and 16.5/inch characters (unjustified).
(Full lines of either width characters will be justified.)
- * Mix normal and **elongated** characters in any format.
(Mixed normal and elongated characters will be justified.)
- * Mix normal and *italic* characters (even *elongated*).
- * Mix normal and **emphasized** characters (**elongated** too).
(This ***one*** is in emphasized, elongated, underlined italics.)
- * Dynamically activate/deactivate **double strike** printing.
- * Dynamically change line spacing (6/inch, 8/inch, 7/72 inch).
- * Set and exercise horizontal tab stops, such as:

Tab 1 Tab 2 Tab 3 Tab 4 Tab n

- * Print block graphics (graphically stated)

Print **BLOCK** graphics.

- * Combine the above demonstrated features in just about any manner they want.

FLEXTXT/80 and SCRIPSIT™ were exercised in composing this page.

FLEXTXT/80 is available now from APPARAT. The purchase price is:

\$34.95

™A registered trademark of The Tandy Corporation.

FLEXTXT/80 (for MX-80 Printers) Requires Graftrax



Apparat, Inc.

"On-going Support for Microcomputers"

4401 S. Tamarac Pkwy. • Denver, CO 80237 • (303) 741-1778 • (800) 525-7674

Scrpsit & TRS 80 are a registered trademark of Tandy Corporation
Freight F O B Denver-call for shipping charges Foreign Orders shipped Air Freight



```

230 PRINT"(C) Change a record"
240 PRINT"(D) Delete a record"
260 PRINT"(T) Sort"
270 PRINT"(F) File names"
280 PRINT"(N) New data file"
290 PRINT"(Q) Quit"
297 PRINT:PRINT NI;" RECORDS FULL, ROOM FOR ";MX-NI-1;" MORE"
300 GOSUB 60000:PRINT
310 IFA$="S"ORA$="s"60SUB2000:GOTO200
320 IFA$="P"ORA$="p"60SUB3000:GOTO200
330 IFA$="A"ORA$="a"60SUB4000:GOTO200
350 IFA$="C"ORA$="c"THENS=3:GOSUB8000:GOTO200
360 IFA$="D"ORA$="d"THENS=4:FS=1:GOSUB8000:GOTO200
370 IF A$="T"ORA$="t"THEN GOSUB7000:GOTO200
380 IF A$="F"ORA$="f"THENGOSUB600:GOTO200
390 IFA$="Q"ORA$="q"ORA$="N"ORA$="n"THENS=0
410 GOTO200
500 IFSS=1THEN530
520 GOSUB2000
530 CLOSE2
540 IFA$="N"ORA$="n"THEN100
550 END
600 "CMD"DIR":GOSUB60000:RETURN
    
```

Load subroutine.

```

1000 INPUT"FILE NAME (NO EXTENTION PLEASE) ";F$
1010 ONERRORGOTO1230
1020 OPEN"I",1,F$+"/HDG"
1030 INPUT#1,NH,NI,MX,LK,FF
1040 ONERRORGOTO0
1130 DIMH$(NH),B$(NH+1),I$(MX),P$(MX),TI$(NH),B$(NH+1,FF)
1140 FORI=0TONH:INPUT#1,H$(I),B$(I):NEXT
1150 INPUT#1,B$(NH+1)
1160 IFNI=0THEN1180
1170 FORI=1TONI:INPUT#1,P$(I):NEXT
1180 CLOSE1
1190 GOSUB20000
1200 IFNI=0THEN1220
1210 GOSUB1300
1220 SS=1:RETURN
1230 PRINT"FILE NOT FOUND";:GOSUB60000:RESUME100
1300 PRINT:FORI=1TONI
1305 PRZ=INT((P$(I)-1)/FF)+1
1310 GET 2,PRZ
1320 I$(I)=FN$B$(B$(CH,P$(I)-FF*(PRZ-1)))
1330 NEXT
1340 RETURN
    
```

Initialize subroutine.

```

1500 INPUT"FILE NAME (NO EXTENTION PLEASE) ";F$
1510 IFF$=""THEN1500
1520 INPUT"How many headings";NH
1530 IFNH<1THEN1520
1540 NH=NH-1:NI=0:LK=0:TZ=0
1560 DIMH$(NH),B$(NH+1),TI$(NH):B$(0)=0
1570 FORI=0TONH
1580 PRINT"Heading #";I+1;:INPUT " ";H$(I)
1590 INPUT"MAXIMUM LENGTH : ";B$(I):TZ=TZ+B$(I)
1610 NEXT
    
```

```

1615 FF=INT(256/TZ):DIMB$(NH+1,FF)
1620 INPUT"WHICH HEADING IS THE LONGEST ON WHICH YOU WILL SORT";
J:J=J-1:IFJ<0ORJ>NHTHEN1620
1630 MX=INT((FRE(A$)-2000)/B$(J)+2)
1640 DIMI$(MX),P$(MX)
1650 GOSUB20000
1660 SS=0:RETURN
1700 PRINT:R=P$(I):GOSUB20100
1710 GET 2,PRZ
1720 FORJ1=0TONH
1730 TI$(J1)=FN$B$(B$(J1,P$(I)-FF*(PRZ-1)))
1740 NEXT
1750 RETURN
1800 PRINT:GOSUB20100
1805 FORJ1=0TONH
1810 LSET B$(J1,R-FF*(PRZ-1))=TI$(J1)
1820 NEXT
1830 PUT 2,PRZ
1840 RETURN
    
```

Write subroutine.

```

2000 PRINT:ONERRORGOTO2290
2010 OPEN"D",1,F$+"/HDG"
2020 PRINT#1,NH,NI,MX,LK,FF
2040 FORI=0TONH:PRINT#1,H$(I);";";B$(I):NEXT
2050 PRINT#1,B$(NH+1)
2060 IFNI=0THEN2270
2070 FORI=1TONI:PRINT#1,P$(I):NEXT
2270 CLOSE1
2280 ON ERROR GOTO0:SS=1:RETURN
2290 PRINT"DISK ERROR":GOSUB60000:RESUME200
    
```

Print subroutine.

```

3000 IFNI=0THENGOSUB9000:RETURN
3005 CLS:PRINT"(A) SCREEN PRINT, DEFAULT FORMAT
(B) SCREEN PRINT, SELECT FORMAT
(C) LINE PRINT, DEFAULT FORMAT
(D) LINE PRINT, SELECT FORMAT:IFPEEK(14312)<>63ANDPEEK(293)<>73
THENPRINT:PRINT"PRINTER NOT READY!":GOSUB60000ELSEGOSUB60000
3006 IFA$="A"ORA$="a"THENS=1:GOTO3040
3007 IFA$="B"ORA$="b"THENGOSUB10000:FS=2:GOTO3040
3008 IFA$="C"ORA$="c"THENS=1:SB=2:LPRINTCHR$(32):LPRINTTIME$:GOTO3060
3009 IFA$="D"ORA$="d"THENGOSUB10000:FS=2:SB=2:LPRINTCHR$(32):LPRINTTIME$:GOTO3060
3010 GOTO3005
3040 SB=1:CLS:PRINT"AFTER EACH RECORD <M> WILL RETURN YOU TO MEN
U":PRINT
3050 PRINT"Press any key . . .":GOSUB60000
3060 GOSUB8010
3090 IFSB=2THENPOKESC,S1:POKESC+1,S2
3100 RETURN
    
```

Print one record to screen.

```

3300 ON FS60SUB 3700,3800
3340 IFSB=1THENGOSUB60000:IFA$="M"ORA$="m"THENS=1
3350 RETURN
    
```

Print one record default format.

```

3700 CLS:PRINT " ":PRINT"RECORD ";I:PRINT " "
3710 FORJ=0TONH
3720 PRINTH$(J),TI$(J)
3730 NEXTJ
3740 RETURN

```

Print one record custom format.

```

3800 CLS:J=1:T=0:B$=""
3820 J1=VAL(MID$(F$(T),J,1)):J=J+1
3830 IFJ1<5THENN=VAL(MID$(F$(T),J,2)):J=J+2
3840 ON J1 GOTO3850,3860,3870,3890,3910,3970
3850 A$=H$(N):GOTO3950
3860 A$=TI$(N):GOTO3950
3870 B$=LEFT$(B$,N-1):IFLEN(B$)<N-1THENFORJ2=LEN(B$)TON-2:B$=B$+
" ":NEXT
3880 GOTO3960
3890 PRINTB$:IFN>1THENFORJ2=2TON:PRINT " ":NEXT
3900 B$="":GOTO3960
3910 IFJ>LEN(F$(T))THENT=T+1:J=1
3920 J2=J
3930 IFMID$(F$(T),J2,1)<>"!":THENJ2=J2+1:GOTO3930
3940 A$=MID$(F$(T),J,J2-J):J=J2+1
3950 B$=B$+A$
3960 IFJ>LEN(F$(T))THENT=T+1:J=1
3965 GOTO3820
3970 PRINTB$:RETURN

```

Add subroutine.

```

4000 SS=0:NI=NI+1
4005 CLS:PRINT"Record";NI:PRINT
4010 FORJ=0TONH
4020 GOSUB4500
4030 NEXTJ
4040 IFLK=0THENR=NI:GOTO4080
4050 R=LK
4060 GOSUB 20100
4070 GET 2,PRZ:LK=CVI(B$(0,R-FF$(PRZ-1)))
4080 GOSUB1800:P%(NI)=R:I$(NI)=TI$(CH)
4090 RETURN
4500 PRINTH$(J);" (";BZ(J);"CHARACTERS";" )";:INPUT " ":TI$(J)
4510 RETURN

```

Change subroutine.

```

5000 CLS:PRINT"(C) Change item, (K) Keep item, or (R) Keep remain-
der of record"
5030 PRINT:PRINT"Record ";I
5040 CS=1:RS=0:FORJ=0TONH
5050 PRINT:PRINT H$(J);" ";TI$(J);" ";
5055 IFRS=1THENPRINT:GOTO5090
5060 GOSUB 60000:IFA$(C)"C"ANDA$(C)"c"ANDA$(C)"K"ANDA$(C)"k"ANDA$(C)"
R"ANDA$(C)"r"THEN5060
5065 PRINTCHR$(ASC(A$)+32$(A$)"Z")
5070 IFA$="K"ORA$="k"THEN5090
5075 IFA$="R"ORA$="r"THENRS=1:GOTO5090
5080 GOSUB4500
5085 CS=0
5090 NEXTJ
5095 RS=0

```

```

5100 IFCS=0:R=PZ(I):GOSUB1800:I$(I)=TI$(CH)
5110 RETURN

```

Delete subroutine.

```

6000 PRINT:PRINT"DELETE THIS RECORD? ";
6070 GOSUB60000:IFA$(Y)"Y"ANDA$(Y)"y"ANDA$(Y)"N"ANDA$(Y)"n"THEN6070
6080 PRINTCHR$(ASC(A$)+32$(A$)"Z"):IFA$="N"ORA$="n"THEN6150
6090 LSET B$(0,PZ(I)-FF$(PRZ-1))=MKI$(LK)
6095 PUT 2,PRZ
6110 LK=PZ(I)
6120 FORI=I+1TONI
6130 I$(I-1)=I$(I):PZ(I-1)=PZ(I)
6135 NEXTI
6140 NI=NI-1:SS=0:I=I-1
6150 RETURN

```

Sort subroutine.

```

7000 IFNI=0GOSUB9000:RETURN
7010 CLS:FORJ=0TONH
7020 PRINT@128+(J*32),"(";J+1;") ";H$(J)
7030 NEXTJ
7040 INPUT "SORT ON WHICH HEADING";J1
7045 J1=J1-1
7050 IFJ1<0ORJ1>NHRETURN
7055 IFJ1<>CHTHENCH=J1:GOSUB1300
7060 PRINT"(A) Ascending or (D) Descending":GOSUB60000
7070 IFA$="A"ORA$="a"THENA=1:GOTO7100
7080 IFA$="D"ORA$="d"THENA=2:GOTO7100
7090 GOTO7060
7100 IFA=1THENFORI=0TONI-1ELSEFORI=1TONI
7110 T=I
7120 FORI1=T+1TONI
7122 PRINT@960,I;I1;
7125 ONAGOTO7130,7140
7130 IFI$(I1)<I$(T)THENT=I1
7135 GOTO7145
7140 IFI$(I1)>I$(T)THENT=I1
7145 NEXTI1
7150 IFT=1THEN7180
7155 FORJ=0TONH
7160 T$=I$(T):I$(T)=I$(I):I$(I)=T$
7170 J1=PZ(T):PZ(T)=PZ(I):PZ(I)=J1
7180 NEXTI
7200 SS=0:RETURN

```

Search subroutine.

```

8000 IFNI=0THEN60SUB9000:RETURN
8010 I1=1:I2=NI:J=0:C1%(0)=-1:BS=1
8015 CLS:PRINT"SEARCH CRITERIA:":PRINT
8020 PRINT" 0 ) RECORD NUMBER"
8030 FORI=0TONH:PRINT@128+((I+1)*32),I+1;") H$(I):NEXTI
8035 PRINT:PRINTNH+2;") BEGIN"
8036 PRINTNH+3;") RETURN TO MENU"
8040 PRINT@896;:INPUT"WHICH FIELD";I:IFI<0OR1>NH+3THEN8040
8045 IFI=NH+2THENC1%(J)=-1:GOTO8150
8046 IFI=NH+3 THEN RETURN
8050 C1%(J)=I-1
8055 CLS:IFI=0PRINT@128,"SELECTING ON RECORD NUMBER":GOTO8060ELS
EPRINT@128,"FIELD SELECTED: ";H$(I-1)

```

```

8060 PRINT@256,"INDICATE (1) SMALLER, (2) EQUAL, OR (3) LARGER"
:GOSUB60000:C2%(J)=VAL(A%):IFC2%(J)<1ORC2%(J)>3THENB060

8080 PRINT@384,"COMPARED TO: ";:IFC1%(J)=-1THENB100
8090 INPUT " ";C%(J):J=J+1:IFJ>7THENB160
8095 GOTO8015

8100 INPUT " ";I:IFI<1ORI>NITHENB100
8110 IFC2%(J)=1THENI2=I
8120 IFC2%(J)=2THENI1=I:I2=I
8130 IFC2%(J)=3THENI1=I
8140 GOTO8015
8150 IFJ<2THENB200

8160 PRINT@896,"1) ITEM MUST MEET ALL CONDITIONS":PRINT@2) ITEM
MAY MEET ANY CONDITION":GOSUB60000:BS=VAL(A%):IFBS<1ORBS>2THENB
160
8200 PRINT"SEARCHING . . .":RS=0:J1=C1%(0):IFSB=2THENPOKESC,P1:P
OKESC+1,P2
8210 DS=0:FORJ=0TO7
8220 IFC1%(J)=-1THENJ=7:GOTO8240
8230 IFJ1<>C1%(J)THENJ1=-2
8240 NEXT

8245 IFJ1>-1 ANDJ1<>CHTHENENCH=J1:GOSUB1300
8246 IFJ1=-2THENDS=1
8250 I=I-1:FORI3=I1TOI2:I=I+1
8251 IFDS=0THENTI$(CH)=I$(I):GOTO8255
8252 GOSUB1700
8255 AS=0:FORJ=0TO7

8260 IFC1%(J)=-1THENJ=7:GOTO8345
8270 ONC2%(J)GOTO8280,8290,8310
8280 IFTI$(C1%(J))<=C%(J)THENB330
8285 GOTO8340
8290 IFTI$(C1%(J))=C%(J)THENB330
8295 IFRIGHT$(C%(J),1)<>"*"THENB340
8298 T=LEN(C%(J))-1:IF LEN(TI$(C1%(J)))<TTHENB340

8302 IFLEFT$(TI$(C1%(J)),T)=LEFT$(C%(J),T)THENB330
8305 GOTO8340
8310 IFTI$(C1%(J))>=C%(J)THENB330
8320 GOTO8340
8330 IFBS=2THENAS=1:J=7
8335 GOTO8345

8340 IFBS=1THENAS=2:J=7
8345 NEXTJ
8350 IFAS=0ANDBS=1THENB355
8352 IFAS<>1THENB380
8355 IFDS=0THENGOSUB1700
8360 IFSB<>3THENGOSUB3300

8365 IFSB=3THENGOSUB5000
8370 IFSB=4THENGOSUB6000
8375 IF RS=1 THEN I3=I2
8380 NEXTI3
8390 PRINT:PRINT"THAT'S ALL":FORI1=1TO500:NEXT
8400 RETURN

```

Error subroutine number one.

```

9000 PRINT"There's no data in memory."
9010 FORI=1TO1000:NEXT:RETURN

```

Print Formatting.

```

10000 IFF$(0)=" "THEN10040
10010 PRINT"SAME FORMAT?":GOSUB60000
10020 IFA$="Y"ORAS$="Y"THENRETURN
10030 IFA$<>"N"ANDA$<>"n"THEN10010
10040 PRINT"(L) LOAD FORMAT, OR (C) CREATE FORMAT":GOSUB60000
10050 IFA$="C"ORAS$="c"THEN10200
10060 IFA$<>"L"ANDA$<>"l"THEN10040
10090 ON ERROR GOTO10170
10100 INPUT"FORMAT NAME: ";A$

10110 OPEN"I",3,A$+"/FMT"
10130 INPUT#3,NF
10140 FORJ=0TONF:INPUT#3,F$(J):NEXT
10150 CLOSE3
10160 ON ERRORGOTO0:RETURN
10170 PRINT"FORMAT NOT FOUND":GOSUB60000:RESUME200

10200 NF=0:J=0:F$(0)=" "
10210 CLS:PRINT"Start in the upper left corner and work across e
ach line"
10220 PRINT"1:HEADING, 2:ITEM, 3:TAB, 4:NEXT LINE, 5:STRING, 6:E
ND":INPUTJ1
10230 IFJ1<1ORJ1>6THEN10220

10240 F$(NF)=F$(NF)+RIGHT$(STR$(J1),LEN(STR$(J1))-1):J=J+1
10250 ONJ1GOTO10260,10260,10300,10300,10350,10400
10260 FORT=0TONH:PRINT@128+(T*32),T+1;" ";H$(T):NEXT
10270 INPUT"WHICH";T:T=T-1:IFT<0ORT>NHTHEN10270

10280 GOTO10310
10300 INPUT"HOW MANY";T:IFT<1ORT>99THENPRINT"OUT OF RANGE.":GOTO
10300
10310 A$=RIGHT$(STR$(T),LEN(STR$(T))-1):IFT<10THENA$="0"+A$
10320 F$(NF)=F$(NF)+A$:A$:J=J+2

10330 GOTO10380
10350 INPUT"STRING: ";A$:A$=A$+"!"
10360 IFLEN(A$)+J>255THENN=NF+1:J=0:F$(NF)=" "
10370 F$(NF)=F$(NF)+A$:J=J+LEN(A$)
10380 IFJ>252THENN=NF+1:J=0:F$(NF)=" "

10390 GOTO10210
10400 INPUT"FORMAT NAME: ";A$
10405 ON ERROR GOTO10460
10410 OPEN"D",3,A$+"/FMT"
10430 PRINT#3,NF:FORJ=0TONF:PRINT#3,F$(J):NEXT
10440 CLOSE3
10450 ON ERROR GOTO0:RETURN
10460 PRINT"DISK ERROR":GOSUB60000:RESUME10400
20000 ON ERROR GOTO 20070
20005 OPEN"R",2,F$+"/DAT"
20010 FORJ=1TOFF
20020 DZ=0

20030 FORI=0TONH
20040 FIELD 2, DZ+(J-1)*TZ AS DU$,BZ(I) AS B$(I,J)
20050 DZ=DZ+BZ(I)
20060 NEXT I:TZ=DZ:NEXTJ
20065 ON ERROR GOTO 0:RETURN
20070 PRINT"DISK OR FIELD ERROR; PRESS A KEY TO TRY AGAIN":GOSUB
60000:RESUME 100
20100 PRZ=INT((R-1)/FF)+1:RETURN
60000 A$="":A$=INKEY$:IFA$=""THEN60000ELSEPRINT:RETURN

```

DOSPLUS

What Does It Mean?

by Alan J. Zett

Now that you've been presented with the pros and cons of DOSPLUS, the next logical step is to gain a better understanding of how to use it. Learning to use the features available in DOSPLUS to their best advantage can make your programming tasks a lot easier and less time-consuming.

Before you can attempt to use any of the new features, a thorough understanding of DOSPLUS responses to operator input must be attained. The most common form of a computer-generated reply is the error message. One of the first things you may have noticed when using DOSPLUS is the use of two-letter abbreviations for all DOS and BASIC error messages.

To avoid any further confusion, this month we present a list of DOSPLUS error messages — explained.

- /0 — Division by zero.
- AD — File Access Denied: Password protected file.
- AO — File Already Open: OPENing or reOPENing a file not previously CLOSED.
- BM — Bad file Mode: Accessing a disk file in a mode other than was specified when OPENed, or an invalid mode type.
- BN — Bad file Number: Accessing a file buffer not previously OPENed, accessing a file buffer with a number higher than was reserved on entry to BASIC.
- BS — Beyond Subscript error.
- CN — Can't CONTinue.
- DD — ReDIMensioned array.
- DF — Disk Full: All free space on disk has been used.
- DS — Direct Statement in file: Attempt to execute or manipulate a non-BASIC disk file from BASIC.
- EF — End of File: End of file was encountered while accessing a disk file.
- FC — Function Call error.
- FD — Bad File Data (on cassette I/O).
- FE — File already Exists: Attempt

to rename a file to the same name as one already in use.

FF — File not Found: An illegal or incorrect filespec was specified, or the file is not contained on the disk specified.

FL — Too many Files: All directory file entries have been used up. Even though there may be enough free space on the disk, there is no room left in the directory.

FO — Field Overflow: More bytes were allocated in a FIELD statement than were specified in the LRL (Logical Record Length — See March, 1982, *SoftSide* for more information) when the file was created or OPENed.

ID — Illegal Direct.

IE — Internal Error: An error in the DOS itself, or a disk I/O error.

IO — Disk I/O error: An error occurred between the disk drive and the computer during a file access.

LS — String Length error.

MM — Mode Mismatch: Attempt to access a file in a form other than was specified when it was created.

MO — Missing Operand.

NF — NEXT without FOR.

NM — Bad file Name: An illegal filespec was specified.

NR — No RESUME statement.

OD — Out of Data.

OM — Out of Memory.

OS — Out of String space.

OV — Overflow error.

RG — RETURN without GOSUB.

RN — Bad Record Number: A record number which is higher than the last record in the file, or a number less than one was specified.

RW — RESUME Without error.

SN — Syntax error.


ST — String formula too complex.

TM — Type Mismatch error.

UE — Undefined Error: This error doesn't exist. Possibly an IE error.

UF — Undefined user Function: An FN statement was encountered before a DEF FN statement was specified.

UL — Undefined Line number.

WP — Write-Protected disk: Attempt to write to a disk that has the write-protect notch covered. 

Your Adventures Will Start Here

Subscribe to Adventure of the Month

The cost? A six-month membership is just \$29 for the tape (\$4.83 per adventure) or \$49 for the disk (\$8.16 per adventure). If you're not sure that you can take six full months of excitement, you can order a single tape for \$7 or a disk for \$10. Or, if you're especially adventuresome, we're offering disks, packed with three great adventures, for only \$26 per disk.

To order, use the card provided in this issue.

Adventure of the Month
4 South Street,
Milford NH 03055

See page 16 for ordering information.



APRIL ADVENTURE OF THE MONTH WITCHES' BREW ADVENTURE

You must find your way to the castle and rescue the Princess who is chained inside its dungeon. A tightly-woven blend of fantasy, horror, and science fiction, this complex adventure will challenge your wits and ingenuity.

Refware THESAURUS

A review by Kathleen E. Boucher

By David C. Whitney (Refware Software Division, David C. Whitney Associates, Inc.) System requirements: 48K TRS-80® Model I or III with two disk drives. Suggested retail prices: *Nouns 1.0*, \$39.95; *Adjectives 1.0*, \$39.95; *Builder 1.0*, \$149.95; each on disk with manual.

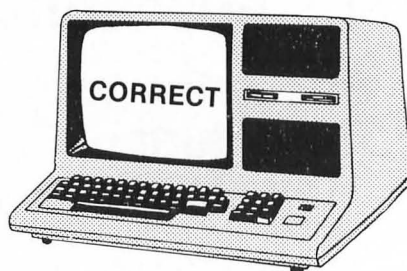
Anyone who has ever had the "pleasure" of writing a college term paper has had ample opportunity to become familiar with *Roget's Thesaurus*. Sometimes it's a real chore trying to find the correct word to fit into the context of your paper. Today, it is much easier to write a paper, letter, or presentation on a word processor, but there is still the fumbling around with the paperback or library edition of *Roget's* work. Now you can update your methods entirely with the *Thesaurus* programs designed to save time and improve efficiency.

Each of the three *Thesaurus* programs comes with a separate manual and a Model I disk. Should you wish to purchase only the manual to explore the possibilities of the programs, they are available separately. Each manual begins with a brief history of the original *Thesaurus* and Mr. Whitney's background. It proceeds in a very easy-to-understand format that requires no programming knowledge on the part of the user. I went through the manuals completely before attempting to run the programs and found that when I actually started to use the programs they were not foreign to me. The manuals are designed so that the top of each page shows the user what the screen display will look like in that part of the program. Below that are full explanations of what this portion of the program accomplishes. I don't believe the manual could possibly be made any easier to follow.

When you order the Model III version the programs are sent on a

Model I disk "to improve the chance that your disk will arrive with no lost data." This was something I was not too familiar with but I now understand that loss of data is less likely on a single-density disk. Converting them to run on a Model III required a bit of time as the files are extensive, but it is worth the trouble. Using a formatted disk, the *Nouns* program

**Right, No Error, Precise,
True, Exact, Perfect,
Strict, Regular,
Melioration,
Rectify...**



uses 112 grams and the *Adjectives* program 113. Unless you have a double-density disk, there isn't any way to append the word lists to be contained on one disk. You must run either *Nouns* or *Adjectives* as the need arises.

THESAURUS: Nouns 1.0 and Adjectives 1.0

When looking for a specific noun (or adjective), the program will search through the 6200-word file (taking about two minutes) until it finds ten related words. If you wish to search for another group of words, you may instruct the computer to do so. You may get 10, 20,

or even 30 alternates for each word you select and you will be told when it cannot come up with any additional ones.

You have the option of having just a listing of the words, or a listing of a sentence ten times in which a substitute word replaces your original word. You can also list your choice to the screen or the printer. I prefer to have a printout with the words in a sentence. Often seeing a sentence containing the word you're looking for is more helpful than the word list alone. My only suggestion for improvement on this option is that the sentences would be easier to read if they were doubled-spaced, especially when your sentence is quite long, as the printout wraps around. But this is a minor inconvenience that I can live with.

Another advantage to the *Nouns* and *Adjectives* programs is the ability to find words that are difficult to spell. If you wanted to use a word but couldn't remember how to spell it accurately, just think of a commonly-used (and easier to spell) synonym, and the computer will come up with the rest.

Both nouns and adjectives must be entered in singular form. If you use a plural the program will not be able to find any substitute. Neither nouns nor adjectives may be any longer than twelve letters or the computer will not accept them and will ask you to try again.

Some words that you may ask the *Thesaurus* to supply may not be accurate substitutes for the word you are looking for. An example is airplane. In this case, some of the words given are more a part of the airplane than an alternate for the word itself (e.g., fuselage, wing, etc.).

With files consisting of 6200 words, I'm sure there will be some I won't find, many I wouldn't need anyway, and the most commonly-used words will be available at my fingertips. I couldn't condemn this



TRS-80®

program for what words may be looking. . . is *Roget's Thesaurus* all-inclusive?

When trying to find a noun with the *Adjectives* program, and vice versa, the program will search, but not finding the word, will ask you to check your spelling. It does not have a default for this, but you know the difference between nouns and adjectives in the first place. I tried this test only to see if the program would choke up on misinformation.

The words in both the *Nouns* and *Adjectives* programs are cross-referenced well. If you get a list of ten words from the program, you could also select one of the ten and then get a listing with your original word included.

THESAURUS: Builder 1.0

The *Builder* program consists of eight separate utility programs designed for and used to construct the *Refware Thesaurus*. Instructions are easy to understand and you need no programming experience in order to build your own specialized thesaurus. With this program, the user can design a thesaurus for a particular field, whether it be medicine, law, business, or any other specialized field.

A blank, formatted disk is used for your words file. It has the capacity for 620 word groups for a total of 6200 words. The first time you use this it will create a file called WORDS. If you want more groups of words, they are added to this file or you may add them to another disk.

Your word files consist of a record number (entry number) and group number. The record numbers are sequential and the group numbers are either 1 or 2 depending on which was entered first. Ten words are in each group, and the length of each word must not be more than 12 letters. If you would like a substitute that contains more than one word such as "set of rules" (for law), you would fill the spaces with a "#" (set#of#rules) and the problem is remedied.

Any user who constructs a specialized thesaurus is asked to submit it to the Refware division for evaluation with the intent of possibly marketing the product commercially with royalties paid to the author.

ALPHFILE automatically creates an alphabetic word file from your *Builder* program. It takes just over ten minutes to create an alphabetic file of 200 words (just over three seconds for each). The file will then break down and have a separate file for each letter of the alphabet and contain an alphabetic listing of your words. The group and record number are also included here in the event that you must make corrections or additions to your file later on.

In READALPH the computer reads the ALPHRED summary and pauses in between each alpha file and between every 55 lines. This is to give you the opportunity to change single sheets of paper if you want.

Disadvantages

If you accidentally end the program before you are actually finished, the program cannot continue with just a CONT command. You must again RUN whatever the program name is. This is only if you instruct it to end. Otherwise, the program cannot ac-

• PRACTICAL • ORIGINAL • SINGULAR • SUPERB • UNUSUAL • FANTASTIC • REMARKABLE • USEFUL • MEANINGFUL • IMPORTANT • SIGNIFICANT • MARVELOUS • WONDERFUL • OUTSTANDING • EXCELLENT • INCLUSIVE • MATCHLESS •

REFWARE THESAURUS

INNOVATIVE READY-REFERENCE PROGRAMS

How many times have you racked your brain for just the right word when writing an important letter, report, or article? How many times have your ideas been misunderstood because the words you used didn't express your thoughts clearly and accurately?

Now, by using the remarkable new **REFWARE THESAURUS** programs, your computer can speedily find those words that are on the tip of your tongue but that you can't quite remember at the moment. And it tells you how to spell them!

Just slip a **REFWARE THESAURUS** disk into your disk drive. Then type in your sentences or paragraphs. The computer will quickly offer a variety of alternatives, retyping your sentences or paragraphs with substitute possibilities chosen from its multi-thousand word vocabulary. It displays the revised sentences on your monitor or types them on your printer as you choose, so that you can mull them over and choose the one that most accurately expresses what you REALLY mean to say.

Having helped thousands of writers learn to express themselves with clarity as editor of such publications as the *World Book Encyclopedia*, the *Encyclopedia Americana*, and the *Reader's Digest Almanac and Yearbook*, David C. Whitney has drawn on decades of editorial experience to prepare the revolutionary **REFWARE THESAURUS** programs, bringing the speed and power of the computer to the aid of anyone who wishes to improve his writing or speaking.

In addition to the specific programs capable of substituting suggested alternate words for nouns and adjectives, **REFWARE THESAURUS Builder** enables engineers, physicians, lawyers, educators, business, physicists, chemists, and other professionals and specialists to develop their own individually tailored vocabularies of hard-to-remember technical words.

REFWARE THESAURUS Adjectives 1.0
6,200 adjectives assist you in choosing the most accurate modifiers in your ads, letters, reports, and speeches.

REFWARE THESAURUS Nouns 1.0
6,200 nouns suggest alternates for the names of persons, places, things, and ideas that you use in your writing and speaking.

REFWARE THESAURUS Builder 1.0
Series of eight utility programs enables the user to develop specialized computer thesaurus disk programs of hard-to-remember technical words and their alternates for personal use. Functions independently of Adjectives and Nouns programs.

REFWARE THESAURUS User's Manuals: Complete descriptions of use of each of the above programs. Included at no additional charge with each of program disks listed above. If ordered separately, price of documentation is refundable upon receipt of order and payment for program disk.

MINIMUM System Required TRS 80* Mod I or Mod III 48K with two disk drives.
*A Trademark of Tandy Corporation

REFWARE* Reference software division David C. Whitney Associates, Inc.
P.O. Box 451, Chappaqua, N.Y. 10514
*A trademark of David C. Whitney Associates, Inc.

Specify Mod I () or Mod III ()

Qty.	Order No.	Title	Unit Price	Total
	1001RT	Adjectives 1.0	\$39.95	
	2001RT	Nouns 1.0	\$39.95	
	5001RT	Builder 1.0	\$149.95	
	6001RT	User's Manual Adj	\$9.95	
	6002RT	User's Manual Nouns	\$9.95	
	6003RT	User's Manual Builder	\$14.95	
				Sales tax (N.Y. state residents only)
				Postage and handling \$ 3.00
Enclosed \$..... () Check () Money Order			TOTAL	\$
Bill: () Visa () MasterCard				
Card #				
Exp. Date Interbank #				
Signature				
Name Phone ()				
Address				
City State Zip				
Mailed First Class, but allow 3 to 4 weeks for delivery, no C.O.D. orders.				

• CLEVER • SUPERLATIVE • FIRST-RATE • WORTHY • SPECIAL • IDEAL • INVALUABLE • SUPER • PRAISEWORTHY • SKILLFUL • NOTABLE • NIFTY • EXPRESSIVE •

identally reboot while running. It cannot be disabled because the BREAK key and the CLEAR key are not operational. Short of rebooting, it will continue on until you give the END command.

Duplicates are not eliminated from the word file. If you use the same word several times, it will show up in your files. Even using the ? for the dummy word will show in your ALPHFILE as "dummy," for whatever amount of times you actually used it.

I get aggravated having to use more disks than necessary. I do not like to change disks constantly in order to use my word file. Using my processor, the *Builder* program, and my words disks amounted to three disks. On a two-disk system, I found I had to keep removing one of them. In order to get past this obstacle, I again played with several of the invisible files. I found that if you copy C5 from the *Builder* program to your words file you will not need to use the *Builder* program to utilize your words file. After getting into BASIC, all you need do is RUN "C5 and your program will

start with the FIND utility of the *Builder* program. From there on, you need only concern yourself with that part. You don't need to swap disks to use your file, only if you wish to get into another part of the *Builder* program.

Each of the manuals advises the user not to try to list or modify the program. Well, no one can tell me not to do something and expect me not to! I immediately tried to list the program. No luck! All I got was some garbled statements which looked more or less like the REM statements in the manual itself. With the help of a couple of programmers and a little ingenuity, we not only got the word files to list, but also to print out. Great! Just what I always wanted! A printout of 6200 words. But wait, I thought. If I look at the printed sheet versus calling up each word to go through the program, I can see what all the words look like.

After carefully looking over many of the alternate words on the listing, I found that although some words don't have the alternates I would like, the majority of them are just

the thing. In comparing the listings to *Roget's* (not each one!), I found that many of them were either in the original *Thesaurus* or adapted from the original.

Although the concepts behind the *Thesaurus* programs are good, their actual operation leaves something to be desired. I found it to be a bit of a nuisance at first to get out of the word processor I was using, load in another disk, reboot and get into BASIC, find what I am looking for, then get back into DOS and my word processor. But then again, what do you do when that trusty thesaurus has been mysteriously removed from your desk?

Once I familiarized myself with the three programs, I actually tried working with them. They were not such a nuisance at all. A minute or two here and there is really worth it to find the appropriate word for your meaning. So, impress people with your unlimited vocabulary through the use of the *Thesaurus* programs. You'll surprise yourself with the amount of knowledge you will pick up as you go along. ☺

Microproof

A review by David A. Kater

by Phil Manfield (Cornucopia Software, Inc., P.O. Box 5028, Walnut Grove, CA 94596) Suggested retail prices:

<i>Standard Microproof:</i>	
TRS-80® Models I and III	\$69.50
TRS-80® Model II	\$149.50
CP/M	\$149.50
Apple	\$69.50
Correcting Feature	\$60.00
Manual only	\$5.00
Patches for <i>Electric Pencil</i> or <i>Scriptit/Superscript</i>	\$35.00
Other patches	no charge

A new breed of software has invaded the word processing scene. The market is flooded with spelling checkers: programs that simplify the process of editing and proofing a text file. Spelling checkers operate by checking each word in a text file against a built-in dictionary. Any word that is not included in the dic-

tionary is flagged as a potential spelling error. In more sophisticated forms, spelling checkers will also make grammatical corrections, hyphenate text, repaginate text, and more.

What a terrific boon for authors, typists, technical writers — anyone who uses word processing software. Imagine the joy of writing without worry, knowing that errors will be brought to your attention at the touch of a button. Well, it's not quite a one-button process, but spelling checkers DO take the drudgery out of editing a document.

Be wary of the advertising hype, though. Spelling checker programs are very useful, but they are not mind readers. They can't automatically change an incorrectly spelled word into the word you REALLY had in mind. And they have limited ability to catch correctly spelled words used in the wrong

context. For example, mistakes like "I told you sew" will go undetected since each word is spelled correctly.

So why should anyone buy a spelling checker? Because these programs can unfailingly catch spelling errors in a few minutes that we might easily miss after countless rereadings. Any word that does not match our customized dictionary of words is brought to our attention for disposition. Spelling checkers can free us to concentrate on manipulating words and organizing concepts rather than fussing with spelling details. Sending a letter to an important business contact? Run it by your spelling checker first. It could save the day.

Microproof

Microproof is one of the premier spelling checker programs for microcomputers and is available in

two versions: *Standard Microproof* and *Correcting Microproof*.

Standard Microproof works with document files created by a word processing program. Misspelled words are listed to the screen or the printer. Corrections can be made using the search feature of the word processor.

Correcting Microproof interacts directly with the word processing program, making it easy to correct errors and get back to writing. With *Correcting Microproof*, the corrections are made automatically and saved to disk after the editing session.

Microproof has been implemented on several different computer systems. This review covers *Correcting Microproof* as implemented with TRS-80® Model I *Scripts* on a two-drive system. Some details are specific to the Model I and *Scripts*, but the overall operation is typical of all forms of *Microproof*.

A User's View

Model I *Correcting Microproof* comes on a single diskette with files on both sides of the diskette. On Side One we find:

ADDTODIC/CMD — Add new words to dictionary
 SPATCH/CMD — Program used to modify *Scripts* or *Superscript*
 CORRECT1/CMD — Correcting program
 CORRECT2/CMD — Correcting program
 MICPROOF/CMD — *Microproof*
 M/CMD — Links MICPROOF/CMD to *Scripts*
 TEST/CMD — Verifies that diskette survived shipment (50 free grans)

On Side Two, there is:

EXAMPLE — Sample text file
 DICT1 — Standard dictionary
 DICT2 — Standard dictionary
 DICT3 — User dictionary (10 free grans)

The first thing to do with the *Microproof* package is backup the diskette. The user must move files around and change some of them; no need to take any chances of los-

ing a file. The original diskette should be stored in a safe place and the backup diskette used as a working master. Single-drive owners must have NEWDOS or access to a two-drive system to make the backups (Cornucopia Software will do this for a nominal charge).

Microproof is supplied with a TEST program to insure that the disk was not damaged during shipment. Use it on the working master.

Using the Program

Now we are ready to proof a document. The operation of *Correcting Microproof* is quite simple. Once *Scripts* has been modified with SPATCH/CMD, we are ready to test out the program; *Microproof* even provides a sample file for our trial flight. We start by loading the file EXAMPLE into *Scripts*. Then activate *Microproof* with BREAK M EXAMPLE. After the file is saved to disk in ASCII format, the user is prompted to insert the dictionary disk. I keep all three dictionary files on the same diskette. If they are stored on separate diskettes, the system asks for them as needed.

The dictionary search is quite fast. Proofing times for typical documents range from 30 seconds to two minutes. This article was checked in about one minute. When the dictionary search is through, misspellings and typos are displayed on the screen. When the document disk is inserted again, *Microproof* takes us through the list one word at a time with the following options:

1. Correct the word immediately:
 A word is corrected by simply retyping it. The corrected text is saved as CORRECT/TXT after we finish with the entire word list.
2. + — Add the word to the dictionary:
 If the word is not really an error, it can be added to the dictionary and optionally coded as a noun (n), verb (v), adverb (a), and/or adjective (j). Sample entries:

+	+ a
+ n	+ j
+ v	+ nvj

The coding option allows for many of the common suffixes without taking up valuable disk space. One problem with the current program: if a word like grans appears on the list, there is no provision for immediately adding the coded root word gran to the dictionary. The user must make a note to manually add it later (see Utility Programs below). Words are added to the dictionary after the user runs through the entire word list.

3. ENTER — Leave the word alone:
 There may be correct words like Phil or Bellevue that we don't want in our dictionary. Skip these words by pressing ENTER.
4. ? — View the word in context (correcting version only):
 If we are not sure just what to do with a word like "sdx," we can request to see it in context. This happens after all the words are accounted for and the dictionary is updated. Hence, the only options for a word seen in context are to correct the spelling or leave it alone.
5. ! — Exit the word list:
 Used to leave the word list before each word is checked.

These five options are easy to use. You can zip right through the word list with very little practice. After the word list is completed, the dictionary is updated and the user is asked to insert the document disk. Words in context are then displayed and corrected as necessary. The corrected document is then saved as CORRECT/TXT and loaded into memory for further editing with *Scripts*.

Dictionary Maintenance

Microproof supports extensive dictionary expansion. The *Microproof* dictionary is composed of three disk files: DICT1, DICT2, AND DICT3. DICT1 and DICT2 contain a standard 50,000-word dictionary. DICT3 is used to add words for special applications. Currently, DICT3 is expandable to as much space as available on a single diskette (about 70,000 words). The

easiest way to use DICT3 is to simply add words as they occur in all word processing endeavors. Heavy users run a small risk of running out of room on the diskette. If this becomes a problem, we can simply keep DICT3 on a separate diskette.

We can even keep a separate DICT3 diskette for each use of *Scipsit*. Sample categories might be: business correspondence, personal correspondence, medicine, engineering, microcomputer articles and reviews, insurance, club newsletter, real estate, law, etc. By keeping separate dictionaries for each of these applications, we can afford room for Aunt Sally in the personal file as well as words like grans in the micro files.

Utility Programs

Cornucopia provides two utility programs to improve our control of dictionary maintenance. The ADDTODIC/CMD program lets us manually add coded or uncoded words to the dictionary. Just create a normal *Scipsit* file of these words and run ADDTODIC. It will add them to the dictionary. A sample file might look like:

```
SALLY
BYTE N
ENCOUNTER VN
FAST VNJ
where N=noun, V=verb, and
J=adjective
```

The second utility program is PRINTDIC/CMD. It transfers all the words in DICT3 to an ASCII format file named DICT3/SRC. This new file can be loaded directly into *Scipsit* and edited like any text file. (The carriage returns loaded in as Ms on my double-density system, but that was easily corrected by saving the file with: S,A DICT3/SRC then reloading it).

The DICT3/SRC file can be edited to your heart's content: delete words, recode words, etc. Then save the edited version as DICT3/SRC. To build a new dictionary, replace the current DICT3 file with the original one from Cornucopia, then run ADDTODIC to add the words from the edited DICT3/SRC file.

The ADDTODIC and PRINTDIC programs give us complete control over the growth of our personal dictionary. If and when DICT3 becomes too unwieldy, we can use *Scipsit* to edit out less-frequently used words or break DICT3 into smaller dictionaries for specialized applications.

Disk Storage

Authors of spelling checker programs for the Model I computer are faced with the problem of limited disk storage. Manfield is to be congratulated for the data compression and coding techniques he used to cram 50,000-plus words into about 68K bytes of disk storage (One CP/M based spelling checker stores 45,000 entries in 136K bytes). We single-density disk users are quite fortunate to have a program with tremendous word storage capacity on our puny systems.

Another master stroke in disk usage is having the program prompt for DICT3 if that file is not found during the dictionary search. This means we can keep DICT3 (our own dictionary) on a separate diskette. If desired, we could even keep a different dictionary for each writing application. So we have virtually unlimited word storage capacity.

There is one feature of the program I would like to see changed: *Microproof's* insistence on creating a duplicate copy of the text file on disk as CORRECT/TXT. This feature forces us to make sure in advance that there is enough room on the document diskette or the system diskette for CORRECT/TXT. A system diskette with the *Microproof* programs leaves about 20 free grans (you can squeeze 25 out of NEWDOS). As an author, I frequently write files that occupy more than 20 grans on the disk. Grrr.

My eventual solution to this dilemma was to designate a separate diskette as a working disk for the text file and CORRECT/TXT. When I wish to check a file, I save it on the working diskette. Using a write-protect tab on the system diskette forces *Microproof* to save CORRECT/TXT on the working diskette. A formatted diskette con-

tains 67 free grans which should be enough space for almost any application. This approach works like a charm with little inconvenience. Problem solved.

Those who have upgraded to double-density operation will wonder what all the squawking is about. Space is rarely a problem with double-density diskettes. And *Microproof* works fine with a variety of the operating systems necessary to use double-density on the Model I.

Limitations and Problems

Spelling checkers are still in their infancy. Even though *Microproof* is very powerful, there are a few changes that would make it more user-friendly.

I am continually confronted with plural forms of words that should have been coded as nouns. My only options are to enter the plural form of these words which takes up space in the dictionary, or let the word stand in the text, then go back later and manually add the properly-coded root word to the dictionary. Why not give me the chance to add the coded root word to the dictionary when the error is first encountered? It would save a lot of headaches.

I would like to see the need for CORRECT/TXT eliminated. Unfortunately, *Scipsit* itself seems to be the source of the problem. It will only operate correctly from a fixed spot in memory, making storage of the entire corrected text in memory impractical.

I could only find a few minor bugs in this program. When viewing a word in context that happens to be in a header or footer, the display goes into double-width, making part of the text unreadable. You can be sure that this will be corrected promptly.

Another problem occurs when no corrections have been made to the text. In this instance, CORRECT/TXT is not a complete duplicate of the text file, but it is loaded into memory at the end of the proofing procedure. A careless user might mistake it for the original text.

Documentation

I found the original *Users Manual* to be the weakest part of the *Microproof* package. Certainly, most of the needed information is in there somewhere, but I didn't enjoy wading through the material that didn't pertain to my system. The current 33-page manual is an improvement. More emphasis is placed on setting up working diskettes and explaining how the program interacts with file storage.

Once I made peace with the file arrangement on the diskettes, I found the manual to be helpful. The manual addresses using *Correcting Microproof*, using *Standard Microproof*, adding words to the dictionary, and deleting words from the dictionary. The Appendices cover: files included with *Microproof*, working disk preparation, patching instructions, potential spelling errors, and examples of proofing a document and expanding the dictionary.

Customer Support

The gestation period of this article was fortunately long enough to witness several important revisions

of *Microproof*. The program has been updated numerous times to respond to new versions of word processing programs, customer requests, and new operating systems.

Current TRS-80[®] versions operate under virtually every known operating system including DBLDOS, LDOS, DOSPLUS, NEWDOS 2.1, NEWDOS 80, and of course TRSDOS. There are versions for *Scripsit* on the Models I, II, and III and for *Superscript*, *NewScript*, *Lazy Writer*, *Electric Pencil*, and CP/M word processors like *Word Star* and maybe even the Apple by the time this article is in print.

Customer requests have prompted the addition of the correcting feature, seeing a word in context, the PRINTDIC program, the ability to edit unlimited pages on Model II *Scripsit*, and even separate manuals for the different versions of *Microproof*. Cornucopia was also most responsive to pleas for help via the phone.

As for their update policy, software revisions are free within one month of purchase and only \$10 thereafter. New manuals are \$5 if there are any pertinent revisions. Cornucopia Software clearly intends to support their customers.

Electric Webster

Speaking of support, Cornucopia has just announced their newest product, *Electric Webster*. It features greater proofing speed, dictionary lookup to locate correct spellings, and optional grammatical checking and hyphenation.

The cost is \$89.50 but it is free to *Microproof* owners who purchased after Dec. 1, 1981, and \$35 to other *Microproof* owners.

Summary

Spelling checkers will change as end-users think up more and more uses for them. We will see improved search times, better handling of homonyms, automatic correction of common mistakes, statistics like total word count and frequency of words, and a host of improvements we can only guess at now.

But there is no need to wait if you have need of a spelling checker now. *Microproof* has taken a bold step toward automating the editing process. With the support offered by Cornucopia and the innovation already evidenced by Manfield, *Microproof* and its offspring promise to continue as leaders in document proofing software. ☺

Hexspell

A review by David A. Kater and David R. Long

by Bernard J. Hughes (Hexagon Systems, P.O. Box 397, Station A Vancouver, B.C. Canada V6C 2N2) Requires 48K TRS-80[®] Model I or III with two disk drives. Suggested retail prices: *Hexspell 2* program — \$99, Manual only — \$12 *Hexspell 1* upgrade — \$35

“What will a computer do?” I asked. The answer came from the salesman without hesitation, “Well sir, it'll do anything you tell it to do!”

His answer was a bit of an overstatement, at least in 1978. The microcomputer revolution was in its infancy. Software was very limited in scope and available only from the hardware vendors. First-generation word processing software was nowhere to be found. Most pro-

grammers were busy cutting their teeth on a better *Star Trek* or *Hunt the Wumpus*.

Microcomputer software has come a long way since then. The challenge of today's micro user is not HOW to acquire software, but WHICH program to choose. The aftermarket has literally exploded with software for business applications. Programming has achieved a high level of sophistication, in some cases rivaling state-of-the-art software on large computer systems.

The *Hexspell 2* spelling checker continues this trend of increasing sophistication and utility. Most importantly, it truly applies the principle for which the computer was devised: to enhance the user's abilities relative to a given task. In this case, the task is proofreading a

document.

Hexspell checks any text file for spelling errors. It alerts the user of potential errors, then saves the corrected file. It provides an extra measure of insurance against shoddy-looking communications.

Documentation

The first thing one discovers about *Hexspell* is, of course, the documentation. Reading the manual is traditionally the most abhorrent part of installing a new program. Computer “experts” can often skip some of the mumbo jumbo; newcomers can't. Fortunately, Hexagon Systems makes the learning process easy for novice and journeyman alike. The 24-page manual is EXTREMELY well

organized and easy to use. We were pleased to find that it clearly explains each step of program use including the process of altering the program for individual needs.

The manual includes sections on using the *Hexspell* program, altering *Hexspell* features with the control file, extending the dictionary or creating a new one with the Clear program, using foreign languages, typical user questions and answers, and an index.

Nearly every section starts with an introduction — very important for first-time users. The manual does a particularly good job of getting the user up and running. It also explains how to use the program for several different system configurations.

Operating Hexspell

The program is activated by typing HEX SPELL. A document is selected for proofing, and the user is prompted to insert the dictionary diskette.

The text scrolls up the screen so that the user can visually review the document right along with *Hexspell*. The user can interactively vary the speed from 0 (peddling as fast as it can) to 9 (dead stop).

Each word in the document file is checked against the roughly 4,000 words in memory and then against the SPELL/LST and SPELL/EXT files respectively on disk. If there is no matching word in the dictionary, then the word flashes on the screen and we are given three options:

- (S) Skip this word and continue proofing.
- (R) Replace this word with a correct spelling.
- (L) Learn the word (i.e., add it to the dictionary).

(S) If the word is so outrageous that you don't intend to use it again (e.g., vavavooooom), just skip the word by pressing S.

(R) If it is clearly misspelled (e.g., compewter), press R then retype the word. *Hexspell* will immediately check this new word to make sure we aren't trying to pull a fast one. If it passes the test, the proofing continues. Otherwise we are given the same three options for the new word. This immediate check is a

neat feature. It allows us to guess at a correct spelling without running to a paper dictionary.

(L) If the word in question is spelled correctly, we can add it to the dictionary by pressing L.

When we are finished checking the document file, the total word count and number of unrecognized words are displayed. If changes were made, the corrected version is automatically copied to the original document. We are then given the option of proofing another document. At the end of the proofing session, the user may delete any words from the dictionary. This means a quick scan of the list of added words that have been accumulating on the printer (one of the print options). The program ends by updating the dictionary files on disk.

Hexspell vs. Batch Programs

Batch word proofing programs operate a bit differently from *Hexspell*. They break the proofing process into two operations. The text file is first checked against the dictionary while the operator takes a coffee break. Then a word list of potential errors is displayed, and the user decides what to do with the flagged words. The words are not typically displayed in context.

The dictionary search is faster in the batch programs, but the overall time to proof a document AND correct errors is about the same for both methods. We found that seeing the text during the dictionary search is preferable to the out-of-context approach used by some spelling checkers.

Bright Student

Hexspell stores up to 4,000 words in memory, and up to 25,000 words in two files on a single-density diskette (more on double-density systems). *Hexspell* uses a unique method of customizing this dictionary to the user's vocabulary. As a document is proofed, frequently-used words are moved toward the front of the dictionary. Unused words work their way to the back. Thus, the dictionary eventually sorts itself in most-used to least-used word order. The more *Hexspell*

is used, the more it adapts the word list to the user's vocabulary. This speeds up the proofing process considerably.

Newly-added words are placed at the front of the dictionary. If the dictionary ever fills up, the least-used words are dropped off the end of the dictionary to make room for new ones. In this way, the 29,000-word capacity will automatically adjust to the user's vocabulary without carrying a lot of dead weight. Users with large vocabulary requirements can get the most out of the storage capacity by keeping separate dictionaries for different word processing applications.

The main disadvantage of this dynamic dictionary approach is one of time. There are plenty of common words not included in the original dictionary. Be prepared to spend time training Fido to speak your language.

Auto Learn

Users with a radical vocabulary including scientific formulas or technical jargon can use the powerful Auto Learn feature. Simply create a file containing a list of new words, and turn on Auto Learn. *Hexspell* will learn every word in the file. Of course, the new word list should be checked carefully before being force-fed into *Hexspell's* dictionary. We don't want to turn out a poorly-educated spelling checker now, do we?

The Auto Learn feature can also be used to teach *Hexspell* a foreign language. The Clear program should be used first to clear out the current dictionary.

Customization

Hexspell's greatest asset is its versatility. With a few keystrokes, we can alter many of the default parameters of the program. Step right up and choose your operating system, or change default values like printer options, alternate character set (for Model III), auto learn (on/off), extended word list (on/off), input file name, and word file name.

We can even change what *Hexspell* does with each of the 256

ASCII codes as they are read from the document file. This feature allows special handling of hyphens, apostrophes, lower case, numbers, formulas, and more. For example, some folks might use a hyphen "-" to underline chapter headings. Standard *Hexspell* will stop at every heading, displaying "-----" as a misspelled word. Not very amusing. But a simple change to the control file causes control file causes *Hexspell* to consider the hyphen as a word separator (like a comma or space) instead of part of a word. That's all it takes.

Problems

The user can give the corrected document any valid file name. This can be a useful feature, but it is too easy to misuse. If drive 0 happens to

be full (and it will be full if you are using the extended dictionary SPELL/EXT) and you forget to add ":1" to the corrected document file name, the program will abort. *Hexspell* hasn't been on the market long enough for bugs like this to be fully worked out.

Hexspell does a tremendous amount of disk access while it is adjusting to the user's vocabulary and word usage. Thus, it is very good at finding weak spots on a diskette. Double-density users would be well advised to use quality diskettes and keep their disk heads clean.

We had trouble getting the extended list (SPELL/EXT) to work reliably for quite some time. It turned out that one particular word inexplicably caused the program to continually abort. But where there is

a will, there is a way. We used the control program to disable the /EXT file, taught the program the problem word, then reenabled the /EXT file. We haven't run into a problem since.

Summary

In *Hexspell 2* we see the computer fulfilling its promise. It completes a task in a matter of minutes that is time-consuming and redundant for us humans. For some reason, the ability to correctly spell even the most common words of the King's English escapes many of us. A properly-tutored *Hexspell* program can free us from that drudgery. *Hexspell 2's* flexibility and ease of use makes it a worthy addition to one's word processing library.

NewScript Version 6.1

A review by Joseph A. Breton

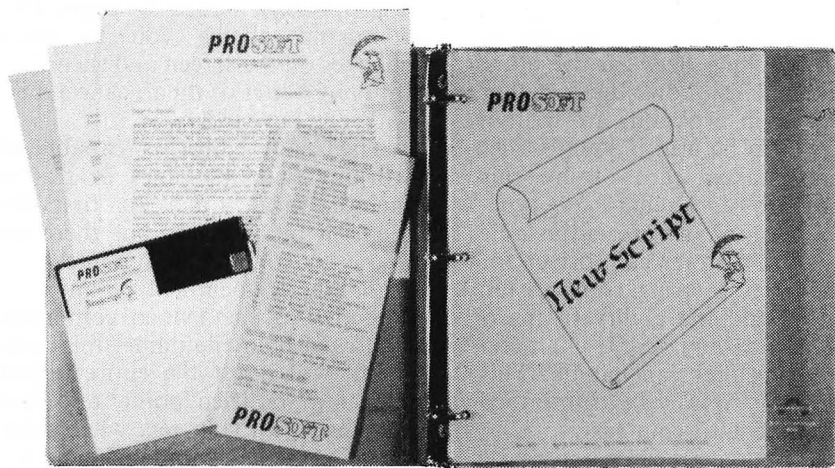
From Prosoft, Box 839, North Hollywood, CA. For TRS-80® Models I and III. Suggested retail price: disk — \$99.95.

The search for the perfect word processing package is never-ending. It's usually necessary to weigh the features you want against the features available on any particular package, and somewhere in there, price plays a major part in the decision. The TRS-80® owner can choose from many packages on the market priced from \$19.95 to \$199.95. *NewScript* Version 6.1 is priced right in the middle at \$99.95.

NewScript 6.1 is supplied in separate but compatible versions for the TRS-80® Models I and III on a ready-to-run TDOS (TinyDOS is a derivative of DOSPLUS) disk. The program is compatible with TRSDOS, DOSPLUS, NEWDOS-80 or LDOS, but there really is no need to transfer the program off of the TDOS disk.

Customizing *NewScript*

The first time you run the program you are presented with a series of menus that allow you to customize your *NewScript* disk. The first menu lists all of the special



printer drivers that are available. The author has thoughtfully provided special drivers for the following printers: Centronics 737/ATARI® 825/RS LP-IV, Centronics 739, Epson MX-80, MX-80 with Graftrax, MX-100, MicroLine-80/82/83, Anadex 9000/9500, Anadex 9001/9501, Modified Selectric Typewriter, RS Daisy Wheel-II, Diablo 1620/C, Itoh Starwriter, and the NEC SpinWriter. This feature allows *NewScript* to use all of the special features that are unique to your particular printer. If your printer is not

on the list, *NewScript* uses a default printer table such as you would find on *Electric Pencil* or *Scriptit*.

(A newer version, *NewScript* 6.2, supports the NEC 8023 dot-matrix printer and the Prowriter. The 6.3 version also allows you to use *Microproof* with the correction feature. Priced at \$149.95, this version fills a complete Model III disk, thus necessitating two disk drives. It is suggested that purchasers return the registration post card; the author periodically offers low-cost updates to the program.)

The second menu of *NewScript*

6.1 allows you to specify how your printer is attached. It supports parallel printers (the assumed default), serial printers using the RS-232 port, and serial printers using the TRS-232 interface from Small Systems Software.

The third menu is to be used should you wish to move *NewScript* to a different Disk Operating System (but why bother?). There are six DOS choices to pick from.

At this point all of your choices are permanently recorded on your diskette for future use. You can, of course, make changes at any time, but in most cases you will not be bothered by these three menus again.

Four Programs in One

As a previous user of *Scipsit*, my first reaction to *NewScript* was — it looks clumsy! The first look through the 180-page manual shows that *NewScript* is actually three separate BASIC programs held together by a fourth program.

NS/CMD is the Machine Language program that handles all of the switching between the other programs. This allows the fact that the two main programs are written in BASIC to be well hidden. Other functions taken care of by this module are a 32-character type-ahead feature, a user-adjustable auto-repeat, a lower-case driver for the Model I, a string compression indicator, and the ability to insert into your text any ASCII character that your printer can handle. This means that *NewScript* can imbed GRAPHICS into your text.

EDIT is used to enter and revise text, as well as save it to disk. A line can have up to 255 characters on it, however, you may view only 60 characters at a time. Unlike *Scipsit*, the screen does not automatically scroll horizontally. If you wish to set up a chart of 132 characters across, for example, you use the View command to specify which 60-character section you wish to see at a particular time. Normally, the text automatically wraps around at the end of the line (at the 60th character or at the end of the word prior to the 60th character). You never have to hit ENTER at the end of a line of text, so you can just keep

on typing. The extended line is only needed to do charts, etc.

In the EDIT mode, the screen is divided into four sections. The top line is called the Command Line, the left-most column is called the LIMA (the Line Manipulation Area), the bottom line is used for a Character Grid (this can optionally be turned off). The middle 14 or 15 lines (depending if you use the Grid) is the text area. This is where most of the action takes place.

The LIMA is a powerful, simple-to-use editor. This editor is very useful for any manipulations that involve the whole line such as deleting one or more lines, inserting one or more lines into the text, moving a line or replicating a line elsewhere in the text.

To use the main editor place commands in the Command Line. There are 56 commands for doing text manipulation from the Command Line. It is possible to do word location, global search and replace, search and delete, etc.

One of my favorite commands is appropriately named Whoops. This command allows you to make changes on the screen and then after thinking better of them, cancel them with one key stroke.

The main editor is capable of working with 400 lines of text in memory at a time. At first this seems a bit limiting, but there are two extra commands that allow you to handle a document of any size. The first is the IMbed command. Followed by a file name, this command will insert the entire named file at a specified point, and then continue with your text when printing. The second command is the Append command. Placing this command at the end of a document followed by a file name, will append the next document without stopping during the printing process.

SCRIPT is the text formatter printing program. You enter almost all of the commands it uses while in the Edit program. The main purpose of the Script program is to format and print a document to your specifications. To do this task, Script uses 45 commands of its own, plus nine additional escape sequences to control your printer. These commands control the page format by margins, justification,

spacing, pitch, automatic indentation, various titles and automatic page numbering.


When Script is running there are an additional 10 run-time options which can make changes just for that one printing session, without saving the changes to disk. These options include double or triple spacing, extra copies, printing just the third page, etc. The program also supplies a unique mini-edit mode that allows you to delete, replace and insert lines during the printing session.

INDEX is a program that will sort (alphabetize) and merge words and phrases that you have marked as key words while creating your document. This feature makes it quite simple to provide easy references for medium to long documents.

NewScript in Use

From the description above, it would seem that *NewScript* is very complicated to use. Not really. In fact, if you do not already have the operating procedures for one or more word processors already committed to memory, I would say that *NewScript* is probably one of the easiest and most logical to use.

The manual has a How-To section which makes it easy to learn how to do: form letters with variables supplied from your mailing list, mailing lists, titles, indices, predefined standard setups, hanging indents and long documents. The power that is in this package can only be equalled by purchasing several software products at a far greater total cost.

I do not consider myself an accomplished author. In fact, the last time I wrote anything like this was probably a high school book report. (I think I got a C+.) But every now and then I come across a product that is great and nobody seems to know about it. The reason for this review is that I use and appreciate *NewScript*. If you are looking for a word processor for your TRS-80®, or you're not satisfied with the limitations placed on you by your present word processor, I firmly believe that you get more than your money's worth from *NewScript*. Thank you, Chuck Tessler, for a great word processor and keep on improving it. 

NEW PRODUCTS

MENLO SYSTEMS

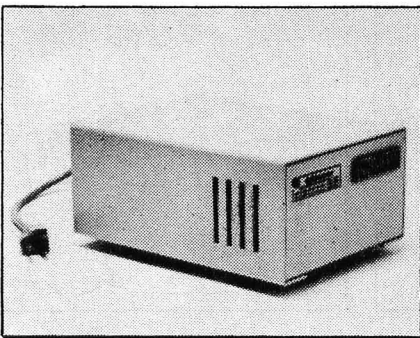
3790 El Camino Real, Suite 221
Palo Alto, CA 94306
(415)327-7424

Autoplot from Menlo Systems opens powerful high-resolution graphics capabilities to the owner of a Radio Shack 48K TRS-80® Model I/III microcomputer with one disk drive. Using an Epson MX-80 or MX-100 printer with GRAFTRAX as output device, this curve-plotting program creates graphs of professional quality, with a resolution of 480 x 192 dots.

Autoplot is written in Disk BASIC, enhanced by fast Machine Language routines in high RAM. Data points are entered with a small user-written BASIC subroutine. A few program lines are sufficient to calculate mathematical functions, request input from the keyboard, or read data from a disk file (i.e., *Visicalc* or *Scriptsit* file.) Many options can be selected with some extra keystrokes. *Autoplot* does all the rest automatically in minutes.

By selecting options, the user can override the automatic scaling of the axes, plot continuous curves or separated markers, choose linear or logarithmic presentation, request a grid overlay, compute derivatives or integrals, plot more than one curve, or select from four different print sizes and formats.

Despite its power and versatility, *Autoplot* is extremely easy to use and comes with a 45-page manual. *Autoplot* is available on disk for \$79.50. A version for the LNW-80 at \$99.50 displays the graph in full resolution on the video monitor, before it is printed.



CUESTA SYSTEMS INC.

3440 Roberto Court
San Luis Obispo, CA 93401
(805)541-4160

Computer Insurance — *The Datasaver* from Cuesta Systems Inc. safeguards the time and information invested in high technology systems by protecting electronic equipment against unexpected power failures or sudden surges that can cause many systems to fail. The unit, priced from \$395, is available immediately.

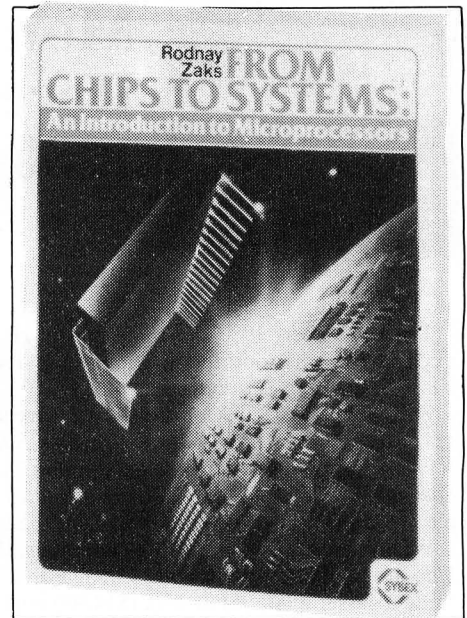
SYBEX

2344 Six Street
Berkeley, CA 94710
(415)848-8233

From Chips to Systems: An Introduction to Microprocessors is a brand-new, beautifully illustrated book written for all owners and users of personal and business computers. This book carries the reader on a fast-paced journey through the world of microprocessors — explaining what a microprocessor is, what it does, and how it does it.

From Chips to Systems, written by Rodney Zaks in a vivid, direct style, will quickly show the reader how easy it is to understand the microcomputer technology which is already changing the shape of the future. Rapid advances in technology have made it possible for all of us to look forward to using a personal computer to enhance and improve the quality of daily life.

This book, priced at \$14.95 is a comprehensive reference tool which is required reading for anyone who wants to have an inside track on the technology which is revolutionizing the future.



In an effort to inform our readers of new products, *SoftSide* welcomes your company's input to this section. Send all information to:

SoftSide Publications 6 South Street Milford, NH 03055

GAMMA SOFTWARE

P. O. Box 25625
Los Angeles, CA 90025
(213)473-7441

Hockey is a high-speed action game for a 16K ATARI® 400/800 and is available on cassette or disk. This Assembly Language game makes use of high-resolution color graphics and generates a variety of sounds including music and crowd noise. It offers nine different game options.

Hockey is played on an enclosed rink, with scoreboard including clock overhead. The game is for two, three, or four players, who use joysticks as controllers. The three-player version is a novelty in which two players team up against the third.

Offensive players carry the puck, pass and shoot. Defensive players steal the puck and intercept passes. Goalies block and clear shots on goal.

An advanced feature of the game is the inclusion of "smart" players who perform automatically. In case of a tie after regulation time, the game goes into sudden-death overtime.

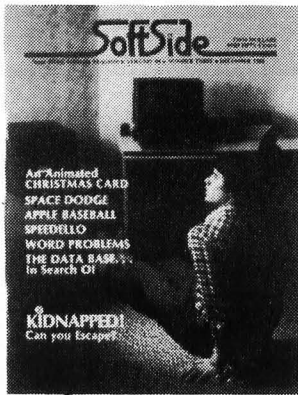
Both cassette and disk versions sell for \$29.95.

THE ALTERNATE SOURCE

1806 Ada Street
Lansing, MI 48910
(517)487-3358

A new software package, the *Automatic Density Recognition* for NEWDOS/80 2.0, solves the number one complaint expressed by users of that popular system — the elimination of manual manipulation of the PDRIVE parameters when using various combinations of single- and double-density diskettes. Interested users may request information on the *Automatic Density Recognition* package by asking for the "DDSD flyer."

DDSD was designed by Allan J. Domuret, noted author for many popular computer magazines. DDSD is no more complicated than executing it from DOS READY. No disk ZAPs of any kind are required. DDSD can even be an "AUTO" file executed every time the user boots the system. This program functions with all TRS-80® DOS systems and will not adversely effect fast CPU clock systems. DDSD will load itself at top of available memory, respecting any and all programs present and is compatible with both Models I and III.



Back Issue of the Month: December 1980

The December, 1980, *SoftSide* boasts 11 feature programs, a couple of short routines, articles and reviews. This issue contains *Developing Database Part 4*.

For the TRS-80®: *Kidnapped*, an adventure; *Missile Evasion*; *Word Problems*; *Command*; and *Space Dodge* whose ATARI® version is also included. Other ATARI® programs: *Speedello*, an Othello-like game; and *States and Capitals*. Apple programs: *Baseball*, *Connect-A-Dot*, and *Christmas Card*.

While they last, copies of the December, 1980, issue of *SoftSide* are available for a mere \$3.50. For a bit more we'll include a cassette (\$9.95) or disk (\$14.95) containing all the programs for your computer that appear in the issue. Remember to specify what computer you use when ordering the magazine and media combinations. Order now and prepare yourself for yet another magazine full of top-notch software for you and your family to enjoy.

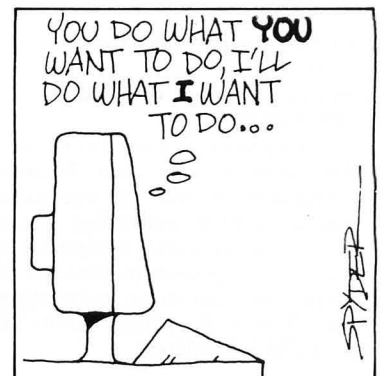
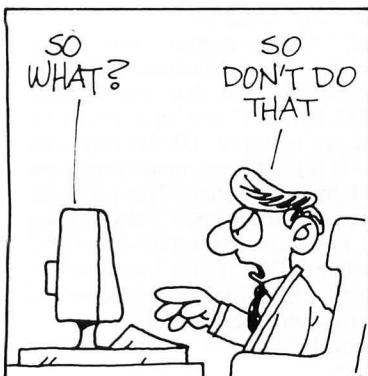
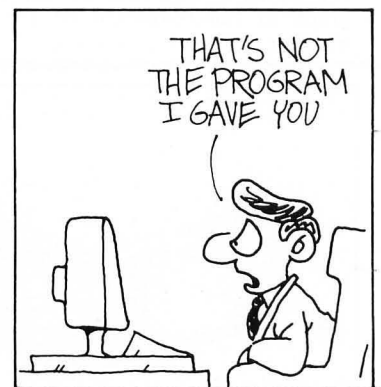
See ordering information on page 16.

Advertiser's Index

Amp Recordings	65
Apparat, Inc	17, 81
Automated Simulations	32
Continental Adventures	60
Diversions	Cover II
Icom	33
National Computer Shows	1
Northeast Expositions	4
Refware	87
<i>SoftSide</i>	10, 85, 96, Cover III
Strategic Simulations	Cover IV
The Software Exchange	20

MACHINE HEAD

BY SPYDER



STOP TYPING!

Get Instant Enjoyment from SoftSide's programs with SoftSide's
Cassette Version (CV) and Disk Version (DV)!

Our media editions let you spend less time TYPING — and more time USING the fine software that **SoftSide** brings you every month. And we let you choose the version you want.

Cassette Version (CV)

SoftSide's Cassette Version (CV) offers you an inexpensive way to enjoy our programs without hours of typing or hunting for errors. All programs are tested and ready to go!

CV gives you the programs offered for your system each month in **SoftSide** on a tape, plus the magazine itself — 12 magazines and 12 tapes per year for just \$75.

Disk Version (DV)

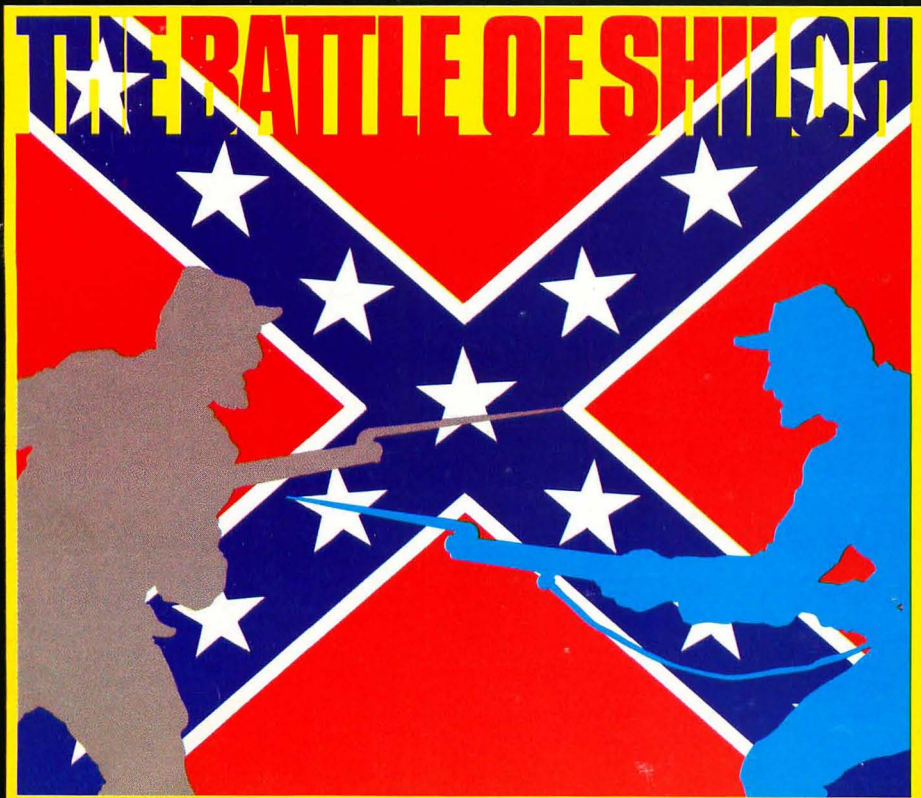
DV contains a BONUS program for your system on the disk in addition to the other programs available that month. Only the documentation for the bonus programs will appear in **SoftSide** magazine, NOT the code. The bonus programs will be of every conceivable type — multiple and Machine Language programs, modified languages, ongoing modular programs and software so extensive, it would take an entire issue of **SoftSide** just to print the code.

Feel like you're missing something? You are. Don't wait to take advantage of our offer — 12 magazines and 12 disks for just \$125 a year. For orders outside the U.S., add \$50. For your convenience we also offer an installment payment plan for MasterCard and VISA holders: Pay just \$32.50 per quarter (a total of \$130 which includes a \$5 billing charge).

To order, use the card provided in this issue.



TWO NEW GAMES FROM SSI FOR THE APPLE® AND THE TRS-80®!



THE BATTLE OF SHILOH: A brigade-level simulation of the first grand battle of the Civil War, pitting the Confederate Army against Grant's troops and Union gunboats.



TIGERS IN THE SNOW: Ghostlike Nazi Tiger tanks and infantry sweep across the dark, frozen forests of the Ardennes against a surprised U.S. force in this division/regiment-level simulation of Hitler's last desperate attack.

We know it hasn't been easy for you TRS-80® owners to see so many great made-for-Apple-only games from SSI pass you by. But then, it hasn't been easy for us to design games for a 16K cassette format good enough to meet our critical standards.

After all, we've got a reputation to protect, a reputation in strategy gaming for unsurpassed sophistication, innovation, realism, and playability.

Well, our designers have been hard at work, and we've not only met but surmounted the challenge. We're delighted to announce two historical wargames — deserving of the SSI label — for both the Apple® and the TRS-80® (16K cassette for the TRS-80 Model I and III; 48K disc for Apple II with Applesoft ROM card).

Combining our extensive wargame-design experience and superior programming techniques, we've given a fresh new look and feel to these favorite classic battles.

At \$24.95 each for TRS-80 cassette and \$39.95 each for Apple disc, these are extraordinary games at quite an ordinary price.

So head on down to your local store and check them out today!

VISA and M/C holders can order by calling 800-227-1617, ext. 335 (toll free). In California, call 800-772-3545, ext. 335.

To order by mail, send your check to: Strategic Simulations Inc., Dept. SS-1, 465 Fairchild Drive, Suite 108, Mountain View, California 94043.

All our games carry a 14-day money-back guarantee.



STRATEGIC SIMULATIONS INC.

As part of our demanding standards of excellence, we use MAXELL floppy discs.

Apple is a registered trademark of Apple Computer Inc.

TRS-80 is a registered trademark of the Tandy Corporation.