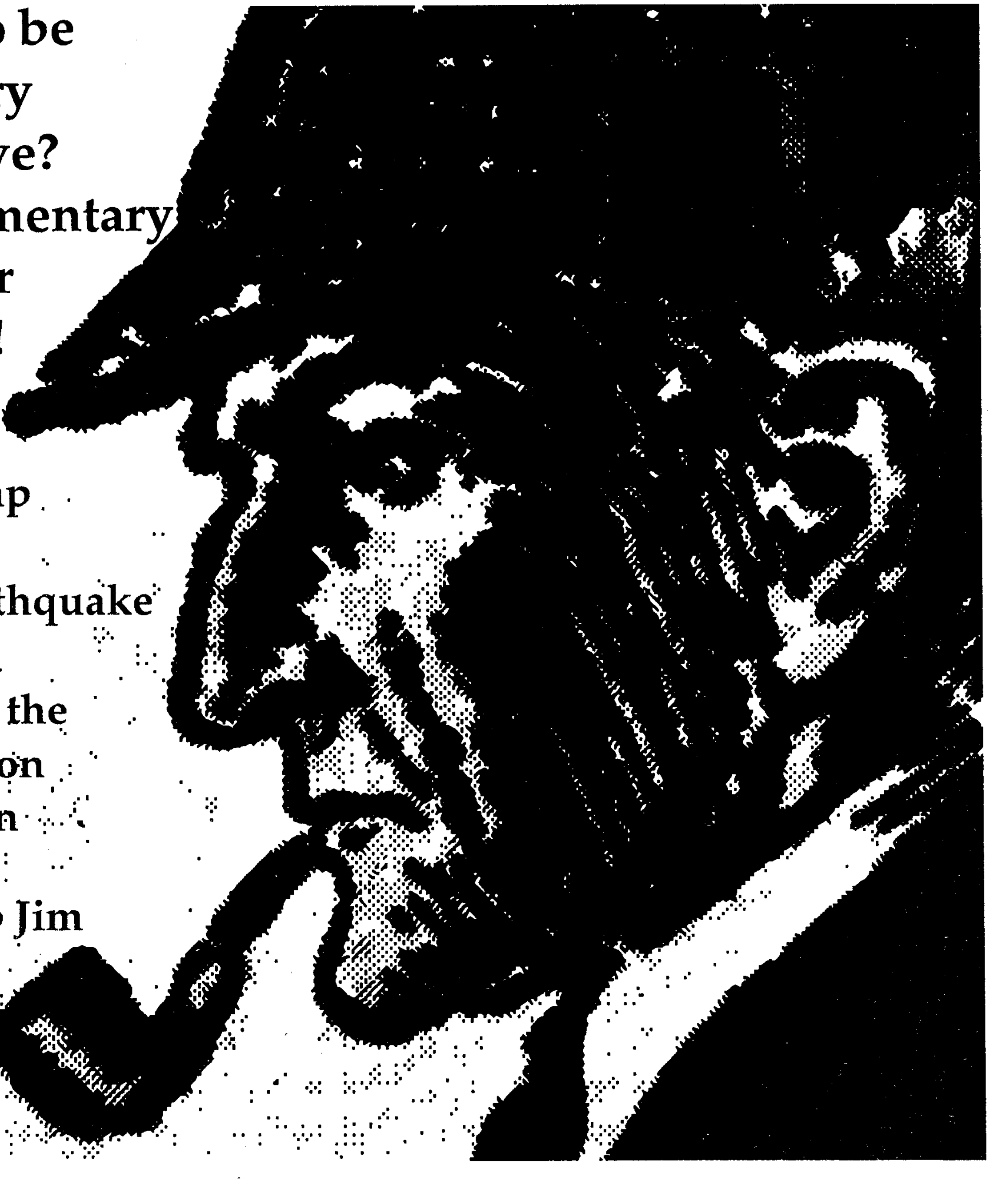
Covering the TI99/4A and the Myarc 9640

MICROpendium

Volume 11 Number 1 February 1994 $3.50



Want to be a literary detective? It's elementary my dear TI user!

See Page 8.

Tex-Comp recovers from earthquake

Figuring the population explosion

Tigercub Jim Peterson dies at 70

CONTENTS

MICROpendium

MICROpendium (ISSN 10432299) is published monthly for $35 per year by Burns-Koloen Communications Inc., 502 Windsor Rd., Round Rock, TX 78664-7639. Second-class postage paid at Round Rock, Texas. POSTMASTER: Send address changes to MICROpendium, P.O. Box 1343, Round Rock, TX 78680-1343.

Foreign subscriptions are $40.25 (Mexico); $42.50 (Canada); $40.00, surface mail to other countries; $52 airmail to other countries.

All editions of MICROpendium are mailed from the Round Rock (Texas) Post Office.

Mailing address: P.O. Box 1343, Round Rock, TX 78680.
Telephone: (512) 255-1512
CompuServe: 75156,3270
Delphi TI NET: MICROpendium
GEnie: J.Koloen
Internet E-mail: jkoloen@holonet.net
John Koloen................................Publisher
Laura Burns....................................Editor

*READ THIS
Here are some tips to help you when entering programs from MICROpendium:
1. Most BASIC and Extended BASIC programs are run through Checksum, which places the numbers that follow exclamation points at the end of each program line. Do not enter these numbers or exclamation points. Checksum is available on disk from MICROpendium for $4.
2. Long Extended BASIC lines are entered by inputting until the screen stops accepting characters, pressing Enter, pressing FCTN REDO, cursoring to the end of the line and continuing input.

TEX★COMP Earthquake Clearance Sale!

ır Northridge, California, warehouse is about five blocks from the center of the January 17 earthquake. While our building stayed up, it has taken us three weeks to pick everything up. If we survived 1983, when TI pulled the plug, we can certainly handle a 6.8 earthquake. There is no better way to see what you are overstocked on than when you have to pick up. To celebrate our survival, we have slashed prices.

There may never be a better time to buy!

Modules 99¢ each—NO LIMIT!

Subject to availability. Effective 3-1-94

PHM 3027	ADDITION & SUBTRACTION I
PHM 3028	ADDITION & SUBTRACTION II
PHM 3093	MILLIKEN DIVISION
PHM 3061	NUMERATION II
PHM 3094	MILLIKEN INTEGERS
PHM 3025	MIND CHALLENGER
PHM 3021	WEIGHT & NUTRITION
PHM 3002	EARLY LEARNING FUN
PHM 3003	BEGINNING GRAMMAR
PHM 3112	PARSEC
PHM 3007	HOUSEHOLD BUDGET MANAGEMENT
PHM 3044	PERS. REPORT GENERATOR**
PHM 3053	TI INVADERS
PHM 3194	JAWBREAKERS II
PHM 3016	TAX INVESTMENT RECORD KEEPING***
PHM 3022	PERSONAL REAL ESTATE
PHM 3023	HUNT THE WUMPUS
PHM 3031	THE ATTACK
PHM 3220	MICROSURGEON
PHM 3057	MUNCHMAN
PHM 3110	CHISHOLM TRAIL
PHM 3218	SUPER DEMON ATTACK
PHM 3006	HOME FINANCIAL DECISIONS
PHM 3035	TERMINAL EMULATOR II
PHM 3052	TOMBSTONE CITY
PHM 3109	LOGO II*
PHM 3055	EDITOR ASSEMBLER*
PHM 3010	PHYSICAL FITNESS

Plus $4 additional shipping & handling

ALSO INCLUDED ARE THE FOLLOWING ATARISOFT MODULES:

DEFENDER, PROTECTOR II, PICNIC PARANOIA

*ADD AN ADDITIONAL $4 .00 EACH ON THESE TITLES DUE TO WEIGHT
**FOR USE WITH PERSONAL RECORD KEEPING ***DISK DRIVE REQUIRED
PERSONAL RECORD KEEPING AVAILABLE AT $4.95

Special Purchase

Personal Record Keeping
PHM 3013
$4.95 (Regular $9.95)

99¢ BOOK BLOWOUT!!! NO LIMIT!

PUBLISHER'S CLEARANCE ON BOOKS FOR THE 99/4A

GAMES TIs PLAY
THE ELEMENTARY TI
COMPUTER PLAYGROUND (TI)
INTRO TO ASSEMBLY LANGUAGE
PROGRAMS FOR THE TI COMPUTER
TI-99/4A GAME PROGRAMS
STARTER PACK I**
STARTER PACK II**
GAME WRITERS PACK I**
GAME WRITERS PACK II**

*Add $3 for Games TIs Play disk or cassette
** Add $3 for disk or cassette for disk or cassette for Starter Pack Game Writer Series.

→ SPECIAL
Kids and the TI — $2.95
The TI in Bits and Bytes with disk — $4.95

$4.95 Per Disk

SPECIAL Buy one, get one free
Any disk or cassette below

EDUCATION

DISKETTE PROGRAMS

PHD 5009	Music Skills Trainer	4.95
PHD 5011	Computer Music Box	4.95
PHD 5018	Market Simulation	4.95
PHD 5030	Speak & Spell II (Ex Basic req.)	9.95
PHD 5031	Speak & Math (TE-II req.)	4.95
PHD 5042	Spell Writer (TE-II req.)	4.95
PHD 5026	Bridge Bidding I	4.95
PHD 5039	Bridge Bidding II	4.95
PHD 5041	Bridge Bidding III	4.95
SPECIAL!!	BRIDGE BIDDING I,II&III	9.95
~~PHD 5020~~	~~Music Maker Demo (use with module!)~~	4.95
PHD 6010	Mystery Melody (stop the music quiz)	4.95

CASSETTE PROGRAMS*
*see disk versions for requirements i.e. TE-II

PHT 6009	Music Skills Trainer	4.95
PHT 6010	Mystery Melody	4.95
PHT 6011	Computer Music Box	4.95
PHT 6018	Market Simulation	4.95
PHT 6031	Speak & Math	4.95
PHT 6042	Spell Writer	4.95
PHT 6026	Bridge Bidding I	4.95
PHT 6039	Bridge Bidding II	4.95
PHT 6041	Bridge Bidding III	4.95
SPECIAL!!	BRIDGE BIDDING I,II&III	9.95
~~PHT 6020~~	~~Music Maker Demo (use with module)~~	4.95

MANAGEMENT AND BUSINESS

DISKETTE PROGRAMS

PHD 5001	Mailing List (upgraded version)	4.95
PHD 5003	Personal Financial Aids	4.95
PHD 5021	Checkbook Manager	4.95
PHD 5022	Finance Manager	4.95
PHD 5024	Inventory Management	4.95
PHD 5027	Invoice Management	4.95
PHD 5029	Cash Management	4.95
PHD 5038	Lease/Purchase Decisions	4.95
PHD 5075	TI/Multiplan upgrade disk	4.95

CASSETTE PROGRAMS

PHT 6003	Personal Financial Aids	4.95
PHT 6038	Lease/Purchase Decisions	4.95

HOME ENTERTAINMENT

DISKETTE PROGRAMS

PHD 5002	TI-TREK (TE-II req. for speech)	4.95
PHD 5010	Mystery Melody	4.95
PHD 5015	Oldies But Goodies I	4.95
PHD 5017	Oldies But Goodies II	4.95
****SPECIAL	Oldies But Goodies I & II	7.95
PHD 5025	Sat. Night Bingo (Ex-Basic & Speech)	4.95
PHD 5037	Draw Poker (Ex-Basic)	4.95

CASSETTE PROGRAMS

PHT 6002	TI-TREK (TEII req. for speech)	4.95
PHT 6010	Mystery Melody	4.95
PHT 6015	Oldies But Goodies I	4.95
PHT 6017	Oldies But Goodies II	4.95
****SPECIAL	Oldies But Goodies I & II	7.95
PHT 6026	Sat. Night Bingo (Ex-Basic & Speech)	4.95
PHT 6037	Draw Poker (Ex-Basic)	4.95

COMPUTER PROGRAMMING AIDS

DISKETTE PROGRAMS

PHD 5007	Teach Yourself 99/4A Basic	4.95
PHD 5019	Teach Yourself Extended Basic	4.95
PHD 5004	Programming Aids I	4.95
PHD 5005	Programming Aids II	4.95
PHD 5077	Programming Aids I,II,III	9.95
PHD 5067	Beginning Basic Tutor	4.95
PHD 5076	Text to Speech (Ex-Basic Speech)	4.95
PHD 5098	TI Forth & manual (Ed/Assem req.)	19.95
PHD 5078	TI Forth Demo Disk (Ed/Assem)	4.95
PHD 5079	TI Forth Source Code (2 disks)	4.95

CASSETTE PROGRAMS

PHT 6006	Programming Aids I	4.95
PHT 6007	Teach Yourself 99/4A Basic	4.95
PHT 6019	Teach Yourself Extended Basic	4.95
PHT 6067	Beginning Basic Tutor	4.95

MATH AND ENGINEERING

specify disk or tape with order

Math Routine Library	4.95
Electrical Engineering Library	4.95
Graphing Package	4.95
Structural Engineering Library	4.95
AC Circuit Analysis	4.95

***SPECIAL--ALL 5 OF THE ABOVE ON DISK OR TAPE---17.95

SPECIAL PACKAGE AND PRICING ON Adventure

THE ADVENTURE SERIES...

SPECIAL PACKAGE AND PRICING ON **THE ADVENTURE SERIES...** ON DISK OR CASSETTE (specify)

THE COMPLETE SCOTT ADAMS SERIES I (ADVENTURELAND, PIRATE ISLAND, THE COUNT MISSION IMPOSSIBLE, VOODOO CASTLE, STRANGE ODYSSEY, MYSTERY FUN HOUSE, PYRAMID OF DOOM, GHOST TOWN, SAVAGE ISLAND I, SAVAGE ISLAND II, AND GOLDEN VOYAGE PLUS KNIGHT IRONHEART BONUS ADVENTURE

NEW LOWER PRICE ONLY $4.95 FOR ALL 13

THE SCOTT ADAMS ADVENTURE SERIES II (THE HULK, SPIDERMAN, BUCKAROO BANZAI, AND THE SORCERER OF CLAYMORGUE CASTLE PLUS 2 BONUS ADVENTURES

NEW LOWER PRICE ONLY $4.95 FOR ALL6

COMMAND MODULE FOR ABOVE ADVENTURES (reg $9.95) **ONLY 95 CENTS** WITH EITHER OF THE ABOVE SERIES

HINT BOOK FOR ALL OF THE ABOVE SCOTT ADAMS ADVENTURES ONLY **95 CENTS** WITH EITHER OF THE ABOVE SERIES

NEW First time on disk — Return to Pirate's Island with Graphics. XBASIC or Editor/Assembler Load **$4.95** Includes disk version of Editor/Assembler

ADVENTURE EDITOR

NOW WRITE AND EDIT YOUR OWN GAMES FOR THE TI ADVENTURE MODULE!

SUPER BUY!

TEX COMP is proud to announce the 99/4A Adventure Editor — an all new program which allows you to write your own adventure programs — Scott Adams move over! This disk or cassette based program allows:

1. Any existing adventure for the Adventure command module to be edited, altered, listed and copied from any storage medium to any other storage medium (i.e. tape to disk or vice versa)
2. New adventure games can be created freely using a template game as a start up step. These games can be conceived using the full power and capabilities of the TI Adventure command module.

The program is written for Mini Memory or Editor Assembler command modules (cassette or disk) and offers a full screen editor and a special easy to use mnemonic language called A.P.L. (Adventure Programming Language) and comes with a complete and detailed 75 page manual.

TEX COMP is also organizing a new TI Adventuring Users Group which will allow low cost exchange of user written adventures between members.

$4.95 Available NOW From TEX-COMP **NEW LOW PRICE**
Specify Mini Memory (Cassette) or Editor Assembler (Disk) versions

****

TEX★COMP
P.O. Box 33084, Granada Hills, CA 91344

VISA and MASTERCARD HOLDERS CALL DIRECT:
(818) 366-6631

TERMS: Add $3 S&H to all orders. California orders add 8.25% sales tax. Canada, Hawaii and overseas shipping extra.

Send $2 for our catalog

AUTHORIZED DEALER

CHARGE IT
WE ACCEPT VISA or MASTERCARD

ORDER BY PHONE 24 HOURS A DAY 7 Days a Week! **(818) 366-6631**

COMMENTS

Memory full

A good many of you recognize the phrase above as not only a computing term, but as the phrase with which Jim Peterson signed off his Tips from the Tigercub columns.

Sadly, we are using it as his epitaph. Jim died last month at age 70. He was active for years in the TI community. Newsletters around the world carried his column. He was a frequent contributor to MICROpendium. Only last month, Charles Good's MICRO-Reviews column included Jim's comments on Don Shorock's Japanese Writing System for the TI.

For years, he refused to hold office in the Central Ohio Ninety-Niners. This was not because he disliked the idea of responsibility, because from all accounts he worked hard in contributing to the club as a member. Rather, his scrupulous character led him to worry about a possible conflict of interest as a software manufacturer. In recent years, however, he was persuaded to share the secretary's job with another member.

In recent years, he was involved less in distributing his own programs and more in distributing public domain and fairware programs at a low cost through his company, Tigercub Software.

We'll miss him.

LITERARY DETECTION

Did William Shakespeare have a hand in the translation of the King James Bible? His name is not on the list of 54 scholars appointed by James I, but some persons think so.

Steven L. Richardson's article on literary detection mentions this and offers a method of literary detection by using a computer to analyze word and phrase frequency. This "Measure for Measure" type of analysis is one of the things a computer works well for, but might not find what some think is a clue that Shakespeare left.

In the King James Bible, count down to the 46th word from the beginning of Psalm 46. Then from the bottom, count up 46 words (not counting the *Selah*, which is not a translated word). The theory is that these words rather than synonyms were selected because the psalm was translated in 1610 by Shakespeare, who was 46 that year.

Of course, the word and phrase analysis has been used extensively by persons seeking to prove that someone other than Shakespeare wrote "Shakespeare."

(And speaking of giving credit where credit is due, we will mention that our rat on the December cover was taken from a drawing by James Grabbs.)

THE SCSI IS HERE

Western Horizon Technologies is introducing a number of new products for TI/Geneve users, not the least of which is its long-awaited SCSI card. Other items include an AT-style keyboard that plugs into the TI console and RXB on a cartridge. The company also has plans for a VGA-compatible 80-column card for the TI and Geneve!

—JK

BUGS & BYTES

WendyXB to show

Though a lot of interest has centered on the proposed introduction of Horizon's SCSI at Fest-West this month, Richard Gilbertson should have his Wendy XB there in cartridge form. The price should be somewhere between $50 and $75.

It's a joke

We won't mention names, because it doesn't matter, but did you hear the one about the TI99/5A?

That's no typo. The TI99/5A is a machine that was supposedly described on CNN. And it has since rebounded across thousands of miles of the Internet as some TI users repeat it without realizing that it was a joke. There is no TI99/5A, which was described as a machine that would be the realization of most user's dreams. And its selling price was quoted as $999.99 and, get this, a TI99/4A could be used as a trade-in valued at $1,050. Now, doesn't this sound too good to be true?

If you know of someone who's offering to buy your used TI99/4As at a good price, tell him gently to save his money. Nobody's going to take them as a trade-in.

Texaments delays

Recent delays in deliveries of products from Texaments have been because of the company's move to Oklahoma, but things are getting back to normal, according to Texaments owner Steve Lamberti. Both Lamberti and Texaments programmer Barry Boone have "day jobs" at Creative Labs now, we hear.

Version 2 coming for MDOS, GPL

We have also heard that Beery Miller is scheduled, all things going well for a release of Version 2 of both MDOS and GPL for the Geneve. V.2 MDOS will reportedly work on both hard and floppy systems, with DSSD required for the floppy drives.

'Tigercub' Jim Peterson dies at 70

James Warren (Jim) Peterson, author of more than 350 programs for the TI99/4A, died Jan. 12 at his home in Whitehall, Ohio. He was 70.

He was retired from the Defense Construction Supply Center. He was a veteran of World War II.

He was the proprietor of Tigercub Software, a company he started 13 years ago after he bought a TI99/4A for his then 16-year-old son. He taught himself programming and produced 90 programs in a year and decided to begin selling them at $3 each. He began a column, "Tips from the Tigercub," as a promotion for his company. The column has appeared in various TI users group newsletters around the world.

Peterson was a founding member of the Central Ohio 99ers. He contributed articles to MICROpendium.

He was also the former secretary of the National Medal Collectors Association and a member of the Central Ohio Cacti and Succulent Society.

He is survived by his wife, Midori Peterson; a daughter and son-in-law, Marianne Mieko and Daniel Ball; a son, Alan Edwin Peterson; two grandchildren, Randy Michael and Scott Alan Ball; a sister, Marjorie Kubitz of Minnesota; and two brothers, Kenneth and Richard Peterson of Minnesota.

Funeral services were held Jan. 15 at the Evans Funeral Home in Columbus, Ohio. Burial was Jan. 17 at Forest Lawn Cemetery in Columbus.

Peterson's son, Alan Peterson, says that TIgercub Software has discontinued business and that outstanding orders will be sent back.

Designer of Geneve 'Swan' among Long Island RR slain

Mi Kyung Kim, age 27, was one of six passengers slain in a shooting rampage on a Long Island Rail Road commuter train Dec. 7.

According to a column by Lucie Dorais in the Ottawa TI99/4A Group's newsletter, Ms. Kim created the Geneve swan and various Myart pictures.

She was born Sept. 30, 1966, in Korea. She worked as an assistant in the mathematics library at Columbia University in New York City.

A wake for her was held Dec. 10 at the Stutzmann Funeral Home in New Hyde Park, New York. Her remains were cremated after a private memorial service.

A Jamaican immigrant, Colin Ferguson, has been charged with 93 counts of murder, attempted murder and assault in the shootings, in which six persons were killed and 19 wounded.

Tex-Comp shook up in California quake

Tex-Comp's warehouse, located in Granada Hills, California, two to three blocks from the center of Southern California's Jan. 17 earthquake, received no structural damage and only minor damage to contents, according to Jerry Price of Tex-Comp.

Price was at his home at the time of the quake. His home was undamaged, but "anything on a shelf or a bookcase ended up on the floor." This included closed cupboard and refrigerator shelves; the doors opened and contents spilled on the floor. Almost all glass articles in the home broke, he says.

As electric power had gone out, he went to his car to listen to reports on the car radio. He then drove to work. On the way, he saw "blocks and blocks of concrete block fences" that had fallen in the quake, as well as numerous collapsed buildings. However, he found the building for the warehouse standing, though everything in it was on the floor.

The company had received a large order from Joy Electronics before the quake and filled it "from the floor," Price says. He says a few monitors and computers broke when they came down from the shelves, but mostly the inventory was undamaged. A "lot of cleaning" was required, but the company has "dug out," he says. "One problem with a company that is mostly mail order is that you have to have a large inventory on hand," he notes.

He notes that the company "took steps" after a tornado took the roof off the building in December 1991 (see January 1992 MICROpendium), and duplicates of its freeware disks are in a safe deposit box. However, no disks sustained damage in the earthquake.

Jerry Price says a few monitors and computers broke when they came down from the shelves, but mostly the inventory was undamaged. A "lot of cleaning" was required, but the company has "dug out," he says.

"It makes you wonder what's next," Price says, noting that his business has been affected by TI's pullout of the home computer market, hit by a tornado, quarantined during the Los Angeles riots after the Rodney King verdict and now hit by an earthquake. "Most businesses don't see that in a lifetime."

FEEDBACK

How to find code for HFDC EPROM

In the article on Page 14 of the January issue, James Lanman says the HFDC EPROM may be type M11. It is undoubtedly an H11. The owner of an HFDC can see the code on the monitor by typing "V" at the first MDM screen after setting the time. This info does not appear in the manual proper but is described in the three-page update dated Oct. 30, 1988.

P.C. Van Nostrand
Seabrook, Texas

Disk inquiry

You gave us a scare in the delayed mailing of the November and December issues. Some of us wondered if this was the beginning of the end!

I have read the subscription form several times and I am still unclear as to what your diskettes cost. I am particularly interested in the assembly programs from No. 1 through the present. Do you have a custom diskette or diskettes which only include a particular subject?

Respond in the magazine as there are others who may have the same question.

Joe N. Simmons
Etowah, Tennessee

First of all, we would not stop publishing MICROpendium without letting our readers know. Second, we have no plans to cease publication.

I assume you're referring to the monthly MICROpendium disks. In order to receive these disks each month, the cost is $40 per year at the beginning of the subscription. Our disk subscriptions run from April through March, so we're near the end of the current 1993-94 subscription period. The 1994-95 series is $40 for U.S. addresses, $41 for Canada and overseas subscriptions are $45 for surface mail and $54.40 for airmail. Sometime around November, we lowered the price of the 1993-94 series to $25. This is because the first eight disks (April through November) can be mailed at one time, instead of monthly. This saves postage and packaging costs.

You've no doubt noticed that our other disk subscriptions for previous years are $25. These disks consist of all the programs that were published during a particular April-March period but do not include all the shareware and freeware programs that we include on the disks that are mailed monthly. The only way to get the monthly disks with the bonus shareware/freeware programs for the 1993-94 series is to subscribe now. After March, we will repackage the 1993-94 series by eliminating the shareware programs. This reduces the number of disks from 12 to 6.

I hope this helps to answer your question.—Ed.

Covina's on line

Your article listing BBS numbers (November 1993) did not include our Club 99 of Covina BBS. If possible, please remind people that we are running at 300/1200/2400 8N1 on a TI99/4A. The number is (818) 339-1134.

Larry Hoffman
Covina, California

Length of stay

Re JK's "Comments," (January 1994), "It's been ten years for us," you state: " ... But I'm happy to be here, and I hope to stay here for a few more years." Why not say: "I'll stay here as long as I can be of service to the TI community"?

I think you (JK) and Laura are doing an outstanding job.

C.J. (Bart) Bartoviak
Cherokee, Alabama

Teaching, learning

One way to pass the past to the future is to attach it to emerging technology. Thus, the heritage of invention (the why) becomes an integral part (how to) of the present functions.

The key to success is not to make the evolution so swift or cumbersome that progress is lost by the necessity of getting this week's job done today.

In the case of the TI99/4A, December's MICROpendium has succinctly captured 10 years of computer revolution. Jerry Price's "why" insight and John Collins "how" system of the month give a current practical application of the TI art and technologies.

I was 63, had a sticky slip stick through my ears and no time when first exposed to the 99ers. The systems popped up often enough since to be used for serious real estate calculations, not knowing it was far superior to Big Blue's and little Mac's. Compaq seems to have succeeded in treading the middle ground to satisfy both leading engineers and society's key manipulators.

Now at 73 we have nieces eager to teach a grandma with time to learn. And grandsons who may invent with grandpa's guidance. I hope both to learn and teach using the heritage that made us great. We can thereby strengthen the technologies that made us strong, to stay strong worldwide in all disciplines.

513-324-2413

Lynn Hokenson
Springfield, Ohio

Christmas specials

I want to pass on some info to you, that should help your advertisers next year obtain the Christmas business they probably didn't get this year.

Being retired, I see our mail carrier pretty much on a regular basis. When my November issue didn't come around the usual time, I asked him about it. He said the following: The Post Office officials had told them to hold all second and third class mail (magazines) until after Christmas. Suffice it to say, I received the November issue Jan. 4 and the December issue around Jan 9. If this was the case nationwide, no one knew about any special offers until it was probably too late. Hope this info has been a help.

Ed Hintermeier
Macon, Georgia

Feedback is a reader forum. The editor may condense excessively lengthy submissions if necessary. We ask that writers limit themselves to one subject per submission. Our only requirement is that submissions be of interest to those using the TI99/4A, the Geneve 9640 or compatibles. Send items to **MICROpendium Feedback, P.O. Box 1343, Round Rock, TX 78680.**

Western Horizon Technologies and Bud Mills Services Present:

SCSI

Small Computer System Interface Hard disk peripheral card

NOW SHIPPING!

Yes, it's finally shipping, our advanced SCSI interface card for the TI 99/4a and Myarc Geneve 9640. For only **$170** (plus shipping and applicable sales taxes) you can add up to 7 SCSI devices to your computer for THOUSANDS of MEGABYTES of storage available at your fingertips.
CALL OR WRITE TODAY for more information.
Users groups:: CALL ABOUT A FREE DEMO UNIT FOR YOUR NEXT MEETING!

ALSO NEW FROM WHT and BMS:

An A-T Keyboard and ROM upgrade for the TI 99/4a.

Now you can add a TRUE A-T keyboard to your TI 99/4a for an unbelievable price of only **$65** (does not include keyboard)
This innovative upgrade installs INSIDE the console and allows the use of BOTH the AT-Keyboard and regular console keyboard!
Also included at **NO EXTRA COST** is an **UPGRADEABLE** 64K of SYSTEM ROM containing O/S bug fixes for popular 80 Column cards like the Dijit AVPC! An optional 0-Wait state 64k memory may be added for future system upgrades and other special software.
ELIMINATE KEYBOARD PROBLEMS TODAY!

RXB IN MODULE FORM! ***** ONLY $60 *****

Now you don't need a Gram Kracker to enjoy this POWERFUL version of TI's Extended Basic. WHT has engineered an inexpensive module to hold RXB. Includes example disk and manual addendums!

Have you been wanting to upgrade your system to a PC power supply? Well wait no more! WHT announces an AT Style expansion box for the masses! Our new P-Box comes in several forms, a basic kit for **$75**, a Baby AT case for **$125**, and a **FULL TOWER** case for **$160**! Our backplane has **10 SLOTS** (only 8 available in Baby AT case) Imagine 9 slots available for your favorite peripherals!

Also coming soon from WHT, **4a MEMEX Jr!** and a **VGA** monitor compatible **80 Column card** for the 99/4a and Geneve! Look for news of these in the coming months or call today for information!

HORIZON COMPUTER

HORIZON 4000 Ramdisk
********NEW LOWER PRICING********
Up to 8 Meg on a single layer Built in RAMBO and PHOENIX

RAMDISK BARE BOARD MANUAL ROS 8.14F $65.00
Zero K kit – above + parts and NO memory "U build" $105.00 Zero K BUILT tested and WARRENTED "U add Memory" $135.00

128k x 8 memory chips $20.00 each
512k x 8 memory are $125.00 each

256k of memory is 2 of the 128k x 8 chips $40.00
1 meg is 8 of the 128k x 8 OR 2 of the 512k x 8 memory
A single layer (16 chips) of 128k x 8 = 2 meg of RAMDISK
YOU can plug in as many chips or as few as YOU want.
P-GRAM for your TI99/4A Grom Emulator for the P-Box

192k Kit $175.00 72k Kit $150.00 add clock $20.00

HORIZON MOUSE $40.00 Mouse Software by Mike Maksimak
Programable to Serial Ports 1-4 avoids Modem Conflicts
Includes Programmers files and helps

Miller Graphic Eproms for CorComp Disk Controllers $35.00

Dijit Ver 2.0 Eprom (solves card conflict with ROS) $5.00

SCSI Controller Card for TI99/4A and Geneve 9640 $170.00

The card is ready and interfaces SCSI Hard Drives, plus some SCSI CD-ROM units and by using an optional FC-1 card can control up to three standard 360k to 2.88meg floppys
Use of the FC 1 card ($100.00) ***REQUIRES*** *a SCSI Controller*

A SCSI controller can operate up to 7 SCSI devices and can co-exist with any existing Hard or Floppy controllers...

ORDER FROM

BUD MILLS SERVICES
166 DARTMOUTH DR
TOLEDO OH 43614-2911
CALL 419-385-5946 6pm - 8pm EST
OHIO add 6.25% tax

WESTERN HORIZON TECHNOLOGIES
10225 JEAN ELLEN DRIVE
GILROY, CA 95020
CALL 408-848-5947
CA add 8.25% tax

Prices subject to change

BBS back on line 419 385-7484 8N1 **VISA and MASTER CARD**

ADD SHIPPING TO ALL ORDERS

UPS Next Day	**$15.00**	**US**	**One fee per shipment**
UPS 2nd Day	**$10.00**	**ONLY**	**US Mail (Priority) $5.00**
UPS 3 Day Sel	**$5.00**	**"**	**Air Mail out of US $10.00**
UPS Ground	**$5.00**	**"**	

Flat rate repair of ANY HORIZON item = $35.00 except MEMORY

WE HARNESS TECHNOLOGY
WESTERN HORIZON TECHNOLOGIES

Extended BASIC

Literary detection with your TI

By STEVEN L. RICHARDSON

When he was about seven years old, my son Ben came up to me as I was reading the newspaper and said, "Dad, our computer is really smart!" "Why do you say that?" I asked. "Because I typed 'How many people are in the world?' and it said '| INCORRECT STATEMENT'." Probably the only answer that might have made the machine look stupid to Ben at that age would have been an "I don't know," which he and I now would recognize as a sign of true computer intelligence.

If there was some interface unit, with software to run it, that permitted the TI to somehow count all the people in the world (perhaps probing the planet from a satellite), Ben would have had a legitimate answer to his question. The computer can come up with the answer if it is given a real method to find it.

Now, I haven't tried this, but I'll suggest a question you might pose to your TI in TI BASIC: "Some have suggested that Shakespeare (who lived from 1564 to 1616) wrote parts of the 1611 King James Bible. Is this true?" I suspect your machine will respond with "| INCORRECT STATEMENT". This answer, which I believe is a correct one, may or may not fit with your opinion, but you need to ask yourself how the computer came up with its response. Somewhere in the dark recesses of its electronic circuitry, does your computer really know who wrote the Bible, or is it simply informing you that what you just typed doesn't make sense?

Since electronic computers became available they have been asked all sorts of questions by all sorts of people. It was probably ten years before I ever touched the keyboard of my first and favorite computer, the TI-99/4A, that I heard frequent repetition of the phrase, "Garbage in, garbage out." The people who were saying these things were speaking of mainframe computers that cost hundreds of times more than the $139.95 that I paid for mine. The phrase is still true, and before we can accept anything a computer has told anyone else, we need to know where their computer got its information and how it processed it, as well as how it was interpreted.

For me, some of the more interesting questions concern the writing style of authors. I find that some writers are easy to read, but others are extremely boring. I find this to be true even if the subject they are writing about is of interest to me. I believe this observation may result from the way the writers construct their sentences, a thing we learn very early in life when we first begin to speak and to write.

Methods for studying writer's style were suggested as early as 1851, but the first major success story happened in 1964, when statisticians Frederick Mosteller and David Wallace studied the 85 Federalist essays, written by Colonial patriots John Jay, Alexander Hamilton, and James Madison. The essays have great historical significance because they urged and eventually persuaded Americans to ratify the U.S. Constitution, which created and empowered the Federal Government. Prior to the Mosteller-Wallace study, the identity of the author of twelve of these essays was undetermined. Based on a computer analysis of the style of authorship, it was shown with a high degree of certainty that James Madison was their author.

Much of the recent development of stylometric methods is due to the efforts of Andrew Q. Morton at the University of Edinburgh, Scotland. As a result of hundreds of text studies in several languages, he has discovered, among other things,

(See Page 9)

TABLE 1: PHRASES

As suggested by Morton, the following is a list of word combinations that he has had good results with, and that you may also wish to test for using the phrases option. Words in capitals should be typed in as is, words in lower case describe the word form you are looking for (adjectives, verbs, etc.) counted manually using "#" to search for them. The symbol "|" represents a word position which may be occupied by any word.

The results from each of these tests should be divided by the wordcount listed in the second column. This kind of processing can easily be accomplished by entering the results and simple formulas into MultiPlan or a similar spreadsheet program.

```
^^A^adj^^^^^^^^^/^A
^^A^|^AND^^^^^^^/^A
^^A^|^OF^^^^^^^^/^A
^^A^|^A^^^^^^^^^/^A
^^A^|^|^A^^^^^^^/^A
^^AND^adj^^^^^^^/^AND
^^AND^THE^^^^^^^/^AND
^^AND^|^OF^^^^^^/^AND
^^AND^|^AND^^^^^/^AND
^^AND^|^|^AND^^^/^AND
^^AS^|^AS^^^^^^^/^AS
^^AS^|^|^AS^^^^^/^AS
^^BE^A^^^^^^^^^^/^BE
^^BY^THE^^^^^^^^/^BY
^^I^AM^^^^^^^^^^/^I
^^I^HAVE^^^^^^^^/^I
^^I^|^I^^^^^^^^^/^I
^^I^|^|^I^^^^^^^/^I
^^IN^A^^^^^^^^^^/^IN
^^IN^THE^^^^^^^^/^IN
^^OF^THE^^^^^^^^/^OF
^^OF^|^AND^^^^^^/^OF
^^AND^THE^^^^^^^/^THE
^^OF^THE^^^^^^^^/^THE
^^IN^THE^^^^^^^^/^THE
^^TO^THE^^^^^^^^/^THE
^^THE^|^AND^^^^^/^THE
^^THE^|^THE^^^^^/^THE
^^THE^|^|^THE^^^/^THE
^^TO^BE^^^^^^^^^/^TO
^^TO^THE^^^^^^^^/^TO
^^TO^|^TO^^^^^^^/^TO
^^TO^|^|^TO^^^^^/^TO
^^YOU^|^YOU^^^^^/^YOU
^^YOU^|^|^YOU^^^/^YOU
^^verb^TO^verb^^/^TO
^^AN^^^^^^^^^^^^/^AN^+^A
^^ANY^^^^^^^^^^^/^ANY^+^ALL
^^NO^^^^^^^^^^^^/^NO^+^NOT
^^UP^^^^^^^^^^^^/^UP^+^UPON
^^WITH^^^^^^^^^^/^WITHOUT^+^WITH
```

STYLOMETRY—

(Continued from Page 8)

that of the 14 books in the New Testament supposed to have been written by the Apostle Paul, only Romans, Corinthians I and II, and Galatians were written by the same person, presumably St. Paul. The others must have been written by someone else. The computer study was unable to determine whether or not Paul had any input into the contents of the remaining documents, and how this information is to be interpreted is open to discussion.

The Book of Mormon is a religious book published by Joseph Smith in 1830, and is believed by Mormons to be the translation of a collection of writings of a succession of ancient American prophets, most of whom are said to have lived between the years 600 B.C. and 421 A.D. In 1979 two Brigham Young University statisticians, Wayne Larsen and Alvin Rencher, claimed that of the four Book of Mormon prophets who wrote multiple blocks of words large enough to be analyzed, each has a style distinct from the others, meaning each varies from the others in his use of what are called noncontextual keywords. The most common words of this sort are listed in line 100 of the accompanying program. Larsen and Rencher explained how they performed their analysis, but unfortunately, as many of you already know, statisticians really aren't very good at explaining how they do things.

Fortunately in 1982 a San Diego computer specialist, Robert L. Hamson, wrote a do-it-yourself book called The Signature of God where he claimed the same results, and using the King James Version of the Bible, he also disclosed his own discovery that the words of Jesus, which are shown in red in some Bibles, have a distinctly different style from that of the Gospel writers Matthew, Mark, Luke, and John, whose writings appear in black type, and who are quoting Jesus. He was able to consistently distinguish samples of text spoken by Jesus from its surrounding Gospel narrative using samples as small as 350 words studying the keywords "AND", "OF", and "TO". Of course, Hamson's computer merely confirmed things that he already believed.

Not to be outdone, an atheist, Ernest H. Taves, wrote a book which he titled Trouble Enough using computer studies to prove to his own satisfaction that the Book of Mormon contains only the writing style of Joseph Smith, confirming, like Hamson, what he already believes.

There are others who have tackled this problem and attempted to approach it from a purely scientific direction, most notably John Hilton and Ken Jenkins, scientists at the University of California, Berkeley. But some of their statistical methods are probably too technical and exacting for beginners like us. But in case you're wondering, they begin with a sample of 4,998 words and slice this into 17 blocks of 294 words each. Each of these blocks is analyzed 49 times rotating through the block in steps of 6 words to provide proper mixing for consistent results. I'm not so sure you and I need to carry our research out to these extremes unless we're trying to prove something to a high degree of statistical certainty to an audience that's inclined not to believe our evidence anyway.

TABLE 2:
Index sentence

In this case, "ss" represents the number of sentences in the sample.

```
A,1^^^^^^^^^^^^^/^ss
AN,1^^^^^^^^^^^^/^ss
AND,1^^^^^^^^^^^/^ss
IN,1^^^^^^^^^^^^/^ss
IT,0^^^^^^^^^^^^/^ss
OF,1^^^^^^^^^^^^/^ss
OF,-1^^^^^^^^^^^/^ss
THE,1^^^^^^^^^^^/^ss
THE,-1^^^^^^^^^^/^ss
WITH,-1^^^^^^^^^/^ss
A,-1^^^^^^^^^^^^/^A
```

The program accompanying this article is a very specialized word processor that utilizes nearly every stylometric method mentioned by Hamson and Taves, and you can use it to either determine which of the above researchers is right, or tackle the question of whether Shakespeare helped write the Bible. If you'd rather avoid religious controversy (and who wouldn't?), you can use the program to study the styles of great writers such as Hesiod, Homer, Goethe, Cervantes, Poe, and yourself. You may wish to discover which of President Clinton's speechwriters wrote a particular

(See Page 10)

BUY - SELL - TRADE
HARDWARE - SOFTWARE
Free! Free!
Call our toll free number and ask for a Free NEW TI Buyers Guide ($2 value)

We will looking for you at Fest West/Tucson Feb 94

Dual ½ HT Kit complete with cables, screws, drill template & instructions $55

800-471-1600
(Nationwide & Canada ORDERS ONLY)

414-672-1600 Local/Tech Support

Dual ½ Ht Ext Kit with case $125

PE Box Complete $85
Corcomp DD Cont $170
Star NX-1001 Printer $165
2400 Baud Modem $69
TI-99/4A Console $35
SC PIO Cables $2
Serial Cables 2 for $1
Modem Cable $15
Banx Box 160 disk holder $15

Corcomp Micro Expansion with 32k, PIO, 4 ½ Height DD Drives in 1 case. 1440k Storage! $350

Huge Genuine TI Inventory Since 1982

Competition Computer
2219 S Muskego Milwaukee WI 53215 Fax 672-8977
Open Daily 9-5 Sat 10-3
Bankcards, Discover, Checks & UPS/COD

STYLOMETRY—

(Continued from Page 9)

speech, or you might want to investigate whether brothers and sisters raised in the same home share a similar writing style, or if a translated document carries the style of the original author or that of the translator.

When you have loaded the program and typed RUN, you are presented with the following menu:

```
    S T Y L O M E T R Y
     1 Create File
     2 Add to File
     3 Load File
     4 Text
     5 List
     6 Edit
     7 Save File
     8 Homogeneity
     9  * Define Block
    10  * Word Lengths
    11  * Sentence Lengths
    12  * Alphabet
    13  * Phrases
    14  * Strings
    15  * Index Sentence
    16 Sort/Count Words
    17 Break
    last choice= 0
    SELECT ONE: _
```

Some people believe the TI is too slow for serious work. But it's only too slow if you are sitting there staring at the screen while it performs its tasks, which can take as long as half an hour. However, if you are away doing something else, the TI may seem too fast. The program has been written to make this latter scenario possible. When you make a selection, a message appears in the lower right portion of the menu screen informing you how long in minutes the activity you have selected will require. If you wish, you can set the time on a kitchen timer and go away, returning to the keyboard when the timer informs you that processing is completed.

Briefly stated, option 1 lets you create a file, typing in each word and pressing enter. The only keys you should use are A to Z, numbers, apostrophes, and periods. You can also use an asterisk to represent words you can't decipher, such as when you are reading handwriting, or when the author leads into a quote of another author (whose writings of course you should not be including in this file). If you want to stop entering text, type "XX" and enter. The program has been dimensioned to allow you to input as many as 1050 words, which is pushing the limits of the TI's memory. Should you attempt to type beyond that limit, the computer will automatically return to the menu.

If you wish to add words to an existing file, option 2 allows you to pick up where you left off.

Option 3, Load File, requires some explanation: Using it you can load a file that has been created using options 1 and/or 2. Actually, options 1 and 2 are not really the best way to input data; an easier method is to type the file into TI-Writer, then by following the instructions at the top of the load screen, the file will be translated into the format used by this program. In doing this, the computer will follow the inputting rules explained above, ignoring all punctuation except apostrophes, periods and semicolons. Apostrophes and periods are left as is, but semicolons are transformed to periods. The entire file is also transformed character by character into uppercase letters. After a file that has been created in this way has been loaded, it should be resaved to disk with another name. You may already have files that you can analyze that were written for other purposes, or there may be files you can download from other devices. The load screen also provides a method that you can merge two files together. More will be said about this later.

Should you wish to review the file, it may be done using options 4 or 5. Of these, the first lets you read each sentence presented as a paragraph. This way you can read what you have written and easily find problems in the text, such as words left out or misspelled. The second method, option 5, lists each word preceded by its word number. If you find either list sweeping past too fast, pressing any key will pause the process; pressing any key again will restart it. When you have seen enough, press "Q" to return to the menu. This instruction also applies to many of the functions described below.

Option 6 gives you limited editing abilities. "S" shows a word you select and several words preceding and following it. "I" inserts a blank position to which a word can be added in a following step; "D" deletes a single word. "E" allows you to edit a word.

Option 7 lets you save to disk a file that you have created or loaded. Files are saved in Display Variable 80 format; loading a file which is the maximum 1050 word size requires about three minutes.

Now let's examine option 8. By its name, homogeneity, it might seem impossibly technical. But if you keep in mind that homogenized milk has been treated so that the cream no longer has the ability to separate, what we are testing for is how similar the file is to itself. The file is sliced up into samples, each initially consisting of 100 words (the size of the samples can be easily modified by holding down "Q" and entering the size you have chosen). A good way to compare the style of two authors is to use option 5 to list the words of the first author down to the end of a sentence at about word number 500. Press "." at which the computer will ask where you want the period to go. Using option 3, a second file can now be added which will start at the ".." and extend to word number 1050 or less. If the two authors exhibit different characteristics in their use of Morton's 20 keywords, or other keywords you may substitute in their place, it will make itself apparent on the graph on your screen in a way that will be obvious if you know how to interpret it. No author will consistently use the words "AND" or "THE" the same number of times for each hundred words he or she writes. The same is true for a thousand words. But the larger the sample the greater the possibility that the counts will vary from each other for that author by only a small number. Two documents written by the same undisputed author should be homogeneous, even when written at different times of the author's life. It may take you awhile to get a feel for how this concept translates out on the CuSum graph. To help you learn what to look for, you may wish to type in and experiment with the following short Extended BASIC program:

```
10 REM CuSum-CuAve Demo !077
20 REM by Steve Richardson !
233
```

(See Page 11)

STYLOMETRY—

(Continued from Page 10)

```
30 REM 25 Oct 1993 !093
40 DATA 10,10,10,10,10,10,10
,10,10 !174
50 DATA 1,9,2,8,3,7,4,6,5 !0
25
60 DATA 1,2,3,4,5,6,7,8,9 !0
25
70 DATA 9,8,7,6,5,4,3,2,1 !0
25
80 DATA 30,31,29,27,30,32,27
,29,28 !230
90 DATA 1,1,1,1,1,6,6,6,6 !0
09
100 DATA 1,2,1,1,2,6,8,6,7 !
014
110 DATA 7,7,7,6,8,1,2,1,1 !
020
120 DATA 9,1,5,2,6,3,7,4,8 !
025
130 DATA -1,-1,-1,-1,-1,-1,-
1,-1,-1 !147
140 V=8 :: H=16 :: N=9 :: CA
LL CHAR(137,"000018242418"):
: CALL CHAR(138,"0000001818"
):: CALL CHAR(136,"0000183C3
C18"):: CALL CHAR(127,RPT$("
1800",4))!112
150 B,E,Z=0 :: CALL CLEAR ::
 DISPLAY AT(2,2):CHR$(138);"
 CuSum":TAB(2);CHR$(137);" C
uAve":TAB(2);CHR$(136);" Bot
h" !245
160 CALL VCHAR(V,H,127,N+2)!
113
170 FOR I=1 TO N :: READ A(I
):: PRINT " ";STR$(A(I));::
B=B+A(I):: C(I)=B/I :: NEXT
I :: D=B/N :: FOR I=1 TO N :
: Z=Z+(A(I)-D) 2 :: NEXT I :
: IF A(1)<0 THEN STOP !130
180 FOR I=1 TO N :: F=C(I)-D
 :: T=H+F :: GOSUB 240 :: CA
LL HCHAR(V+I,T,137):: NEXT I
 !086
190 FOR I=1 TO N :: E=E+A(I)
-D :: T=H+E :: GOSUB 240 !18
4
200 CALL GCHAR(V+I,T,L):: IF
 L=137 THEN CALL HCHAR(V+I,T
,136):: GOTO 220 !049
210 CALL HCHAR(V+I,T,138)!19
0
220 NEXT I :: DISPLAY AT(20,
1):"|Mean=";STR$(INT(B/N|10+
.5)/10):"|Sdev=";STR$(INT(SQ
R(Z/(N-1))|1E4+.5)/1E4)!225
230 CALL KEY(0,K,S):: IF S T
HEN 150 ELSE 230 !083
240 IF T<1 THEN T=1 !077
250 IF T>32 THEN T=32 !184
260 RETURN !136
```

When the above program is run, each time you press enter you will see nine numbers displayed on the screen, they are then plotted on a graph showing the CuSum or Cumulative Sum, as developed by Morton, also the CuAve or Cumulative Average. These two numbers are equal at the beginning and end of the plot, but often take different routes as the plot progresses, and always end on zero. About the fifth time you press enter you should be seeing the type of graph that indicates different authors, first an ideal case, and then what you are more likely to encounter. You should also feel free to try the program on sets of numbers of your own choosing. If the authors are different in their use of the words "OF" or "TO", for example, there will be a dramatic change in that graph beginning at the point where the second file was merged to the first. This shouldn't be expected to happen with every word, but you should make a note of words for which it does happen when comparing two different authors. The program also has been instructed to provide you with the average of all the samples, called Mean, and the Standard Deviation, called SDev. When you run option 8, to keep things simple, the numbers are not projected to count per thousand. Most of the tests described below will be.

Further tests can be conducted on a single file by slicing it up into units, or blocks, then comparing them to each other by using the tests preceded on the menu by asterisks. Option 9 lets you determine which portion of the file you wish to test. Regardless of the block size you select, each stylometric measurement, with the exception of alphabet (which is in percent), and sentence length (in words per sentence) will be projected to count per thousand. After a block size has been selected, the number of words in the block will be added to the menu display to keep you informed. The results of the following tests can in most cases also be output to a "PIO" printer by following the instructions at the bottom of the screen prior to the return to menu. As stated above, holding down "Q" through a cycle will give you the option to terminate most tests.

Option 10 plots a graph of word lengths, with a unit consisting of one character. This is based on the theory that some writers prefer to use longer words than others. The larger the sample, the more bell-shaped the curve described by the graph will be. You will also note you are given the opportunity to change block size in this and the following tests by simply typing in a range of word numbers. For example, entering "51-250" sets the block size at 200 words, with word 51 being the first and 250 being the last. The smaller the block you are testing, the less bell shaped the graph will be.

Option 11 plots a graph comparing sentence lengths. This has been a successful test for writings in ancient Greek and some other languages, but it appears that we writers of modern English are not so consistent in our use of this indicator. Maybe we need to rethink what defines a sentence: does it begin as a thought is introduced and end when the writer proceeds to another subject? In English that seems to better describe a paragraph.

Option 12, Alphabet, simply counts the percentage of the sample used by each letter of the alphabet. Often-repeated keywords will have an effect on the ratio of letters contained in them, as opposed to other alphabetic letters.

Option 13 deals with phrases that may reflect an author's word habits. The computer will search the document for two words of your choosing which occur either next to each other or separated by a fixed number of words. Any phrase that you suspect is repeated often by a writer can be located using this option. As an example, you might be looking for "AND^|^|^AND". The search will present you with each time "AND" is separated by two other words. Some advanced tests will require you to identify words, such as adjectives or adverbs occurring immediately behind a word, such as "THE". To do this, you simply need to count occurrences as they march across your screen. Pressing

(See Page 12)

STYLOMETRY—

(Continued from Page 11)

the space bar after every ten counted helps you keep track; the computer will later ask you how many you counted and project that number to a thousand. Table 1 lists example phrases to search for.

Number 14, Strings, lets you search for occurrences of an alphabetic letter or combination of letters and, if you wish, replace them with another string. This may be useful for searching for words containing certain combinations of letters, such as "ION", "ING", or other style indicators. This option also allows you to painlessly replace abbreviations such as "MR." and "MRS." with "MR" and "MRS", removing false sentence endings from the file.

Option 15, Index Sentence, lets you search for occurrences of words in a certain position in the sentence. Some writers may frequently begin their sentences with "AND", or use "THE" as the second to the last word. This option permits you to test for these and other features. The method you are to follow in doing this is demonstrated each time you select this option. Table 2 lists some tests you may want to try.

Although you should save a file as soon as it has been created, none of the above tests destroy the continuity of the file. The preceding statement isn't true of the next option, 16, so you should be sure you have saved the file before you try to Sort/Count. The program begins by sorting the entire word list into alphabetical order, using a Shell sort; the number of occurrences of each word is then counted and the count is tacked to the beginning of each string, which is then sorted again. The result is a listing of each word, preceded by the number of times per thousand it appears in the document. You are also informed how many words were introduced, which indicates how large the vocabulary of the author is. When the process of sorting and counting is done, the list is paraded past you on the screen. If you wish to stop the listing, press any key. If you want to see it start listing again, press "Q". Other options will then be presented to you. Unlike the methods used by the other researchers, this program has been written to conserve paper. In printing the list it will format the report to usually one page of paper (in doing this, some of the longer words may be truncated).

The final option, 17 Break, allows you to stop from processing to perform a SIZE command, or quit altogether. In addition to informing you of the amount of memory used, the SIZE command forces the computer to perform a garbage collection, which frees up memory, and permits the program to subsequently run faster. You will also note that the program displays an inventory of available RAM each time it returns to the menu.

Results from the stylometric tests you will have performed can be recorded on a form of your own design, then transferred to MultiPlan and analyzed using the methods described in the two tables, fed into a CuSum program, plotted using The Missing Link, or evaluated by any other procedure you may think of. I suggest you study the works of writers such as Hamson and Taves for ideas, or curl up with a good statistics textbook.

And if you discover that Shakespeare really did write the Bible, before you call a news conference, first carefully check your numbers to be certain you won't be embarrassed later. You probably need to sample several blocks each totalling 5000 or more words. The source for each text should be carefully identified. Once you have done this, consider the following: do all undisputed samples give the same results? Are there observations that you ignored (even subconsciously) that don't match to your conclusions? Are there freak events in the sample that may give misleading results? Was your analysis carefully and critically done? These are important questions that should be taken seriously. After all, we now live in a computer age. Anybody with a TI can check your work.

STYLOMETRY

```
10 !************ !123
20 !*STYLOMETRY* !003
30 !************ !123
40 !by Steve Richardson !178
50 !22Jan94 !109
60 !!131
70 !~~~~~~~~~~~~~~~~~~~~~~ !
087
80 W=1051 :: DIM D$(26),A$(1
051)!211
90 !Morton's Keywords: !078
100 DATA 20,THE,AND,OF,TO,IN
,A,AN,BE,I,YOU,NO,NOT,WITH,W
ITHOUT,ALL,AS,BY,ANY,UP,UPON
,!056
110 CALL LOAD(-31878,0):: CA
LL CHAR(128,"00"&RPT$("F",14
)):: CALL CHAR(137,"00001824
2418"):: CALL CHAR(138,"0000
001818")!253
120 CALL CHAR(136,"0000183C3
C18"):: CALL CHAR(127,RPT$("
1800",4)):: CALL PEEK(-31888
,K,D):: IF K=59 AND D=227 TH
EN 150 !003
130 CALL CLEAR :: PRINT "You
 must first execute:": : : :
 : : : : : : : :"CALL FILES(
1)": :"RUN" :: END !063
140 W$,N$,C$,F$,M$,F,P,G,V,Z
,O,B$,S,Y,B,X,A,N,C,I :: CAL
L SCREEN :: CALL COLOR :: CA
LL KEY :: CALL SOUND :: CALL
 HCHAR !205
150 !@P- !064
160 DATA 0,0,3-18,0-3.5,0-5,
,2,.5,0,2-2,0,2-32,0-1.5,2-1
,0-1,31,0 !006
170 F$="words" :: M$=RPT$("=
",28)&"Press <ENTER> to do i
t again or any key and <ENTE
R> to  return to menu" !036
180 ON ERROR 180 :: CALL PEE
K(-31974,A,B):: PRINT " ";A*
256+B-2487;"VDP bytes free"
:: RESTORE 160 :: FOR I=1 TO
 17 :: READ D$(I):: NEXT I !
203
190 CALL SCREEN(12):: CALL C
OLOR(13,4,1):: CALL COLOR(14
,7,1):: CALL CLEAR :: CALL K
EY(3,K,S)!034
200 DISPLAY AT(2,5):"S T Y L
 O M E T R Y": :TAB(6);"1 Cr
eate File":TAB(6);"2 Add to
File":TAB(6);"3 Load File":T
AB(6);"4 Text" !117
210 DISPLAY AT(8,6):"5 List"
:TAB(6);"6 Edit":TAB(6);"7 S
ave File":TAB(6);"8 Homogene
ity":TAB(6);"9  * Define Blo
ck":TAB(5);"10  * Word Lengt
```

(See Page 13)

STYLOMETRY—

(Continued from Page 12)

```
hs" !052
220 DISPLAY AT(14,5):"11  *
Sentence Lengths":TAB(5);"12
  * Alphabet":TAB(5);"13  *
Phrases":TAB(5);"14  * Strin
gs" !031
230 DISPLAY AT(18,5):"15  *
Index Sentence":TAB(5);"16 S
ort/Count Words":TAB(5);"17
Break": :"last choice=";Z ::
 IF Y=0 THEN 240 :: DISPLAY
AT(22,20):Y-X+1;"wds" !127
240 RESTORE :: DISPLAY AT(24
,1):" SELECT ONE: " :: ACCEP
T AT(24,14):Z :: IF Z=6 THEN
 960 !180
250 IF Z<1 OR Z>17 THEN 190
:: DISPLAY AT(24,1):"IS";Z;"
CORRECT (Y/N)" :: W$="("&D$(
Z)&"m)" :: DISPLAY AT(23,29-
LEN(W$)):W$ !043
260 ACCEPT AT(24,21):W$ :: C
ALL CLEAR :: IF W$<>"Y" THEN
 190 !236
270 FOR I=1 TO 17 :: D$(I)="
" :: NEXT I !246
280 ON Z GOTO 310,300,620,38
0,370,960,940,2150,290,1380,
1500,1080,490,1080,2000,1590
,2120 !217
290 PRINT F$,"1-";STR$(N): :
 :"last choice=",STR$(X);"-"
;STR$(Y): : :,:: INPUT "":W$
 :: GOSUB 2340 :: GOTO 180 !
074
300 N=N+1 :: FOR C=N-6 TO N-
1 :: PRINT TAB(8);:: PRINT U
SING "####  "&A$(C):C :: NEX
T C :: GOTO 320 !000
310 CALL KEY(5,K,S):: N=1 ::
 PRINT "SOURCE:": : :: LINPU
T "":A$(0):: PRINT : :!164
320 CALL KEY(3,K,S):: DISPLA
Y AT(15,9):"'XX' to quit" !1
44
330 ON ERROR 330 !084
340 FOR N=N TO W :: PRINT TA
B(8);:: PRINT USING "####  "
:N;:: INPUT "":A$(N):: IF N=
N OR A$(N)="XX" THEN 360 !14
5
350 NEXT N !228
360 N=N-1 :: X=1 :: Y=N :: G
OTO 180 !020
370 CALL CLEAR !209
380 PRINT : : : :A$(0): : ::
 FOR I=X TO Y :: CALL KEY(0,
K,S):: IF S=0 THEN 410 !125
390 IF K=81 THEN 480 !026
400 CALL KEY(0,K,S):: IF S=1
 THEN 410 ELSE 400 !187
410 IF K<>46 THEN 440 !180
420 PRINT : : :: INPUT "Add
period at Word #: ":I :: A$(
I)=A$(I)&"." :: I=I-23 :: IF
 I>0 THEN 470 !046
430 I=X-1 :: GOTO 470 !195
440 IF Z=4 THEN 460 !223
450 PRINT USING "####### "&A
$(I):I :: GOTO 470 !230
460 PRINT A$(I);" ";:: IF PO
S(A$(I),".",1)THEN PRINT : :
!122
470 NEXT I !223
480 PRINT : : :: INPUT M$:W$
 :: IF W$="" THEN 370 ELSE 1
480 !196
490 B=0 :: FOR I=12 TO 26 ::
 D$(I)="0" :: NEXT I :: PRIN
T "phrase:": : : :: CALL HCH
AR(1,1,32,128)!254
500 DISPLAY AT(1,1):"Use * f
or non-phrase words;":" # so
 you can count verbs,":"  ad
jectives, nouns, etc." :: IN
PUT "":W$ :: K=LEN(W$):: FOR
 I=1 TO K :: G=POS(W$," ",1)
:: B=B+1 :: IF G THEN 520 !1
38
510 D$(I)=W$ :: D$(I+13)=STR
$(K-LEN(W$)+1):: GOTO 530 !0
86
520 D$(I)=SEG$(W$,1,G-1):: D
$(I+13)=STR$(G+VAL(D$(I+12))
):: W$=SEG$(W$,G+1,K):: NEXT
 I !126
530 FOR I=1 TO B :: DISPLAY
AT(22,VAL(D$(I+12))+1):STR$(
I);:: NEXT I :: PRINT : : ::
 INPUT "keyword a: ":O :: B$
=D$(O):: DISPLAY AT(23,12):B
$ :: INPUT "keyword b: ":G :
: C$=D$(G):: DISPLAY AT(23,1
2):C$ :: D=G-O :: K=B :: FOR
 I=1 TO B !239
540 IF D$(I)="*" OR D$(I)="#
" THEN K=K-1 !147
550 NEXT I :: N$=C$&"." :: O
=O-1 :: V=Y+1-X :: PRINT :TA
B(8);V;F$:TAB(4);"#" :: C=0
:: FOR I=X TO Y-D :: IF A$(I
)<>B$ AND B$<>"#" OR A$(I+D)
<>C$ AND A$(I+D)<>N$ AND C$<
>"#" THEN 600 !152
560 FOR G=I TO I+D-1 :: IF P
OS(A$(G),".",1)THEN 600 !035
570 NEXT G :: PRINT USING "#
### ":I-O;:: FOR A=0 TO B-1
:: PRINT A$(I+A-O);" ";:: NE
XT A :: PRINT :: C=C+1 :: CA
LL KEY(0,F,S):: IF S=0 THEN
600 !135
580 IF F=32 THEN PRINT SEG$(
M$,1,28)!077
590 CALL KEY(0,F,S):: IF S<>
0 THEN 590 !080
600 NEXT I :: PRINT :: PRINT
 K;F$:C;"counted" :: IF B$="
#" OR C$="#" THEN INPUT "how
 many did you count? ":C !12
9
610 O=(C*1E3/V*10 AND-1)/10
:: PRINT C;:: PRINT ,"freq="
;O: : :: INPUT M$:W$ :: CALL
 HCHAR(21,1,32,96):: IF W$="
" THEN 490 ELSE 1480 !009
620 ON ERROR STOP :: C$="" :
: C=0 :: W$="Precede Filenam
e with '" !119
630 DISPLAY AT(2,1):W$&"+' t
o":" merge a 2nd file after
'..'": :W$&"@' if you are lo
ading a TI-Writer or ASCII f
ile.": :W$&"#' to":" load TI
-W ignoring numbers." !012
640 DISPLAY AT(14,10):"To st
op processing,":TAB(11);"hol
d down any key." !034
650 DISPLAY AT(22,1):"*LOAD
FILE*": : :: INPUT "Device N
ame:":N$ :: W$=SEG$(N$,1,1):
: IF W$<>"@" AND W$<>"#" AND
 W$<>"+" THEN 700 !011
660 N$=SEG$(N$,2,LEN(N$))::
IF W$="#" OR W$="@" THEN 750
 !138
670 FOR I=1 TO N-1 :: IF POS
(A$(I),"..",1)THEN 690 !097
680 NEXT I !223
690 C=I+1 :: C$=A$(0)&" <to
"&STR$(N)&">; " :: IF I<N TH
EN A$(I)=SEG$(A$(I),1,LEN(A$
(I))-1)!129
```

(See Page 14)

STYLOMETRY—

(Continued from Page 13)

```
700 OPEN #1:N$,INPUT :: IF C
 THEN LINPUT #1:A$(0):: PRIN
T : :C$:A$(0):: A$(0)=C$&A$(
0)!128
710 FOR N=C TO W-1 :: CALL K
EY(0,K,S):: IF S THEN 900 !0
53
720 LINPUT #1:A$(N):: IF EOF
(1)THEN 910 !183
730 NEXT N !228
740 N=N-1 :: GOTO 900 !110
750 FOR I=0 TO N :: A$(I)=""
 :: NEXT I :: CALL KEY(5,K,S
):: PRINT : :"SOURCE:": : ::
 LINPUT "":A$(0):: PRINT ::
N=1 :: OPEN #1:N$,INPUT !184
760 LINPUT #1:B$ :: CALL KEY
(0,K,S):: IF S OR EOF(1)THEN
 900 !061
770 IF LEN(B$)=0 OR ASC(B$)=
46 OR POS(B$," .",1)OR POS(B
$,"    ",1)THEN 760 !159780
B$=B$&" " :: FOR C=1 TO LEN(
B$):: A=ASC(SEG$(B$,C,1))::
IF A>47 AND A<58 THEN 790 EL
SE 800 !209
790 IF W$="#" THEN 890 ELSE
880 !091
800 IF A>64 AND A<91 THEN 88
0 !148
810 IF A>96 AND A<123 THEN A
=A AND 95 ELSE 830 !072
820 GOTO 880 !194
830 IF A=33 OR A=59 OR A=63
THEN A=46 !115
840 IF A=42 OR A=46 OR A=39
THEN 880 !123
850 IF A>43 AND A<46 THEN A=
32 !131
860 IF A<>32 THEN 890 !105
870 GOSUB 920 :: GOTO 890 !0
57
880 A$(N)=A$(N)&CHR$(A):: IF
 A=46 THEN GOSUB 920 !014
890 NEXT C :: N$=SEG$(N$,2,L
EN(N$)):: GOTO 760 !034
900 A$(N)="" :: N=N-1 :: IF
POS(A$(N),".",1)=0 THEN 900
!182
910 X=1 :: Y=N :: CLOSE #1 :
: GOTO 1480 !174
920 IF LEN(A$(N))=0 THEN 930
 :: PRINT USING "##### "&A$(
N):N :: N=N+1 :: IF N=W THEN
 900 !061
930 RETURN !136
940 ON ERROR STOP !216
950 PRINT "*SAVE FILE*" :: P
RINT :: INPUT "Device Name:
":N$ :: OPEN #1:N$,OUTPUT ::
 FOR I=0 TO N :: PRINT #1:A$
(I): :: NEXT I :: CLOSE #1 :
: GOTO 1480 !081
960 PRINT "*EDIT*": :"  S=S
how":"  I=Insert":"  D=Del
ete":"  E=Edit": : : :: PRI
NT "COMBINE FUNCTION+WORD NU
MBER": : :: INPUT " (EXAMPLE
 D372): ":W$ !136
970 IF LEN(W$)<2 THEN 960 ::
 A=VAL(SEG$(W$,2,LEN(W$)))::
 W$=SEG$(W$,1,1)!178
980 IF W$="S" THEN 1060 !121
990 IF W$="I" THEN 1020 !070
1000 IF W$="D" THEN 1040 !08
6
1010 IF W$="E" THEN 1050 ELS
E 960 !110
1020 FOR I=N TO A STEP -1 ::
 A$(I+1)=A$(I):: NEXT I :: A
$(A)="" :: IF N=W-1 THEN 106
0 !179
1030 N=N+1 :: GOTO 1060 !014
1040 FOR I=A TO N :: A$(I)=A
$(I+1):: NEXT I :: A$(N)=""
:: N=N-1 :: GOTO 1060 !029
1050 PRINT :A;A$(A):: PRINT
A;:: IF A>0 THEN INPUT "":A$
(A)ELSE LINPUT "":A$(0)!175
1060 IF A<3 THEN A=3 !043
1070 Y=N :: PRINT : : : :: F
OR I=A-3 TO A+3 :: PRINT USI
NG "##### ":I;:: PRINT A$(I)
:: NEXT I :: PRINT : : :: IN
PUT M$:W$ :: CALL HCHAR(21,1
,32,96):: IF W$="" THEN 960
ELSE 1480 !085
1080 GOSUB 1340 !145
1090 C$=STR$(O)&" characters
" :: PRINT TAB(6);C$: : :: I
F Z=12 THEN 1130 !195
1100 INPUT "string= ":B$ ::
IF W$<>"R" THEN 1120 !103
1110 PRINT "Replace '";B$;"'
 with: ";:: INPUT "":N$ !031
1120 PRINT TAB(5);"#" :: GOT
O 1180 !019
1130 PRINT TAB(13);"....1...
.2....3." :: FOR D=65 TO 90
:: C=0 :: B$=CHR$(D):: CALL
KEY(0,K,S):: IF S THEN 1270
!133
1140 FOR I=X TO Y :: B=1 !11
3
1150 A=POS(A$(I),B$,B):: IF
A=0 THEN 1170 !123
1160 B=A+1 :: C=C+1 :: GOTO
1150 !208
1170 NEXT I :: GOTO 1240 !14
0
1180 C=0 :: FOR I=X TO Y ::
B=1 !237
1190 A=POS(A$(I),B$,B):: IF
A=0 THEN 1230 !183
1200 B=A+1 :: PRINT USING "#
#### ":I;:: PRINT A$(I);:: I
F W$<>"R" THEN 1220 !1571210
 A$(I)=SEG$(A$(I),1,A-1)&N$&
SEG$(A$(I),A+LEN(B$),LEN(A$(
I))):: PRINT " => ";A$(I);!1
08
1220 PRINT :: C=C+1 :: GOTO
1190 !152
1230 NEXT I !223
1240 F=(LEN(B$)*C/O*1E3 AND
1)/10 :: CALL SOUND(10,310,0
):: PRINT B$;:: IF F THEN PR
INT USING "=#### ##.#%":C,F;
ELSE PRINT "=   0    0%";!18
4
1250 PRINT RPT$(CHR$(128),F/
2 AND-1):: IF Z<>12 THEN 127
0 !065
1260 D$(D-64)=B$&RPT$(" ",5-
LEN(STR$(F)))&STR$(F):: NEXT
 D !006
1270 IF W$="R" THEN B$=N$ !0
29
1280 PRINT :SEG$(M$,1,56):"'
P'=Print 'R'=Replace"&SEG$(M
$,57,27);"return";:: INPUT "
":W$ :: CALL HCHAR(21,1,32,9
6):: IF Z=14 THEN 1300 !049
1290 IF W$="P" OR W$="" THEN
 1320 ELSE 1480 !076
1300 IF W$="R" THEN 1110 !17
0
1310 IF W$="" THEN 1090 ELSE
 1480 !090
1320 OPEN #1:"PIO" :: IF W$=
"P" THEN K=1 ELSE K=0 !138
1330 FOR S=0 TO K :: PRINT #
S:"ALPHABET COUNT":" ";A$(0)
```

(See Page 15)

STYLOMETRY—

(Continued from Page 14)

```
:" ";C$: : :: FOR I=1 TO 13
:: PRINT #S:"  ";D$(I);"%","
 ";D$(I+13);"%" :: NEXT I :
: NEXT S :: CLOSE #1 :: GOTO
 1270 !013
1340 V=Y+1-X :: DISPLAY AT(1
2,9):"Please Wait" :: G=0 ::
 O=0 :: FOR I=X TO Y :: C=LE
N(A$(I)):: IF POS(A$(I),"*",
1)THEN 1370 !043
1350 IF POS(A$(I),".",1)THEN
 C=C-1 !235
1360 IF C>G THEN G=C !198
1370 O=O+C :: NEXT I :: CALL
 CLEAR :: RETURN !030
1380 GOSUB 1340 :: PRINT "Ma
x=";G: : : :: PRINT TAB(8);"
Word Lengths": :TAB(9);V;F$:
TAB(8);:: PRINT USING "###.#
#=ave":O/V :: PRINT : :: FOR
 A=0 TO G :: CALL KEY(0,K,S)
:: IF S THEN 1450 !070
1390 P=0 :: FOR I=X TO Y ::
C=LEN(A$(I)):: IF POS(A$(I),
"*",1)THEN 1420 !033
1400 IF POS(A$(I),".",1)THEN
 C=C-1 !235
1410 IF C=A THEN P=P+1 !143
1420 NEXT I :: CALL SOUND(10
,600,0):: F=(P/V*1E3 AND-1)/
10 :: D$(A)=RPT$(" ",4-LEN(S
TR$(A)))&STR$(A)&RPT$(" ",5-
LEN(STR$(F)))&STR$(F)&"%"&"
"&RPT$("#",F)!086
1430 IF F THEN PRINT USING "
## #### ##.#%":A,P,F;ELSE PR
INT USING "##    #    #%":A,
P,F;!128
1440 F=F/5 AND-1 :: PRINT RP
T$(CHR$(128),F):: NEXT A ::
PRINT !008
1450 PRINT SEG$(M$,1,34);"'P
' to print";", block,":SEG$(
M$,57,50);": ";:: INPUT "":W
$ :: CALL CLEAR :: IF W$="P"
 THEN 1490 !036
1460 I=ASC(W$):: IF I>47 AND
 I<58 THEN 1470 ELSE 1480 !0
92
1470 B$=W$ :: GOSUB 2340 ::
GOTO 1380 !216
1480 CALL SOUND(100,310,0)::
 GOTO 180 !255
1490 OPEN #1:"PIO" :: PRINT
#1:"WORD LENGTHS":A$(0):TAB(
9);V;F$:RPT$(" ",6);(O/V*1E3
 AND-1)/1E3;"=ave": : :: FOR
 A=0 TO G :: PRINT #1:D$(A):
: NEXT A :: CLOSE #1 :: GOTO
 1990 !048
1500 PRINT TAB(6);"Sentence
Lengths": : :: OPEN #1:"PIO"
 :: IF W$<>"P" THEN 1520 !04
0
1510 PRINT #1:"SENTENCE LENG
THS":A$(0): :!147
1520 O=0 :: P=0 :: FOR I=X T
O Y :: B=1 :: FOR C=I TO Y :
: A=POS(A$(I),".",1):: IF A
THEN 1540 !230
1530 B=B+1 :: I=I+1 :: NEXT
C !229
1540 CALL KEY(0,K,S):: IF K=
81 THEN 1570 !158
1550 PRINT USING "###":B;::
PRINT RPT$(CHR$(128),B/3 AND
-1):: P=P+B :: O=O+1 :: IF W
$="P" THEN PRINT #1:TAB(4-LE
N(STR$(B)));STR$(B);" ";RPT$
("#",B)!035
1560 NEXT I !223
1570 B$="Ave. Sentence="&STR
$((P/O*100 AND-1)/100)&" Wor
ds" :: C$="S./1000 words="&S
TR$((1E5/P*O AND-1)/100):: P
RINT :B$:C$ :: IF W$="P" THE
N PRINT #1:B$:C$ !078
1580 PRINT :: CLOSE #1 :: PR
INT SEG$(M$,1,34);"'P' to pr
int":SEG$(M$,57,50);:: INPUT
 "":W$ :: CALL CLEAR :: IF W
$="" OR W$="P" THEN 1500 ELS
E 1480 !089
1590 ON ERROR STOP :: A=0 ::
 GOSUB 1820 :: CALL SOUND(10
0,110,0,350,0,500,0):: PRINT
 TAB(11);"counting": : : : :
 : : : : : : : :: B=X :: C=1
 :: V=Y-X+1 :: FOR I=X TO Y
!109
1600 IF POS(A$(I+1),".",1)TH
EN A$(I+1)=SEG$(A$(I+1),1,LE
N(A$(I+1))-1)!137
1610 IF (A$(I)<>A$(I+1))THEN
 1630 !062
1620 C=C+1 :: GOTO 1650 !072
1630 A$(B)=STR$(C/100)&" "&A
$(I):: IF B<>I THEN A$(I)=""
 !080
1640 B=B+1 :: C=1 :: DISPLAY
 AT(24,1):Y-I !241
1650 NEXT I :: Y=B-1 :: GOSU
B 1820 !107
1660 CALL CLEAR :: FOR I=X T
O Y :: A=LEN(A$(I))+1 :: C=P
OS(A$(I)," ",1)+1 :: S=(VAL(
SEG$(A$(I),1,C-1))*1E6/V AND
-1)/10 :: PRINT TAB(13-LEN(S
TR$(S)));S;SEG$(A$(I),C,A-C)
!230
1670 CALL KEY(0,K,S):: IF S=
0 THEN 1700 !238
1680 IF K=81 THEN 1710 !237
1690 CALL KEY(0,K,S):: IF S=
1 THEN 1700 ELSE 1690 !217
1700 NEXT I !223
1710 I=X+V-1 :: PRINT "<";ST
R$(X);"-";STR$(I);">":V;"Wor
ds Analyzed":" (projected to
 1000)":Y-X+1;"words introdu
ced": :!072
1720 PRINT SEG$(M$,1,28):" t
o display press ENTER":"   t
o print press  'P'":"   to q
uit  press  'Q' ":"   keywor
ds press  'K'";:: INPUT "":W
$ :: CALL HCHAR(20,1,32,128)
!209
1730 IF W$="" THEN 1660 !127
1740 IF W$="P" THEN 1770 !06
2
1750 IF W$="K" THEN 1910 !19
8
1760 IF W$="Q" THEN 1990 ELS
E 1660 !232
1770 OPEN #1:"PIO" :: I=X+V-
1 :: PRINT #1:"*WORD COUNT*"
:A$(0):"<";STR$(X);"-";STR$(
I);">";V;"Words Analyzed (pr
ojected to 1000)";Y-X+1;"wor
ds introduced)": :!096
1780 D=INT((Y-X+1)/5):: FOR
I=X TO D :: FOR P=0 TO 4 ::
K=D*P+I+X-1 :: C=POS(A$(K),"
 ",1)+1 :: S=(VAL(SEG$(A$(K)
,1,C-1))*1E6/V AND-1)/10 !04
4
1790 D$(P)=RPT$(" ",6-LEN(ST
R$(S)))&STR$(S)&" "&SEG$(A$(
K),C,10):: IF P=0 THEN D$(P)
=SEG$(D$(P),1,15)ELSE D$(P)=
SEG$(D$(P),2,15)!068
1800 NEXT P :: PRINT #1:D$(0
```

(See Page 16)

STYLOMETRY—

(Continued from Page 15)

```
);TAB(16);D$(1);TAB(32);D$(2
);TAB(48);D$(3);TAB(64);D$(4
):: NEXT I :: FOR I=D*5 TO Y
-X !109
1810 C=POS(A$(X+I)," ",1)+1
:: S=(VAL(SEG$(A$(X+I),1,C-1
))*1E6/V AND-1)/10 :: PRINT
#1:TAB(69-LEN(STR$(S)));STR$
(S);" ";SEG$(A$(X+I),C,10)::
 NEXT I :: PRINT #1:"" :: CL
OSE #1 :: GOTO 1720 !232
1820 CALL SOUND(100,150,0,40
0,0,750,0):: PRINT TAB(11);"
sorting": : : : : : : : : :
: : :: G=Y*1.5 !185
1830 G=INT(G/2):: DISPLAY AT
(24,1):G :: IF G=0 THEN 1900
 !145
1840 FOR C=X TO Y-G :: D=C !
191
1850 K=D+G :: IF A$(D)>=A$(K
)THEN 1890 !238
1860 GOTO 1880 !174
1870 IF A$(D)<=A$(K)THEN 189
0 !022
1880 W$=A$(D):: A$(D)=A$(K):
: A$(K)=W$ :: D=D-G :: IF D>
0 THEN 1850 !040
1890 NEXT C :: GOTO 1830 !21
5
1900 RETURN !136
1910 IF LEN(D$(1))=0 THEN 19
40 !096
1920 FOR I=1 TO O :: PRINT D
$(I)!201
1930 CALL KEY(0,K,S):: IF S
THEN 1930 ELSE NEXT I :: GOT
O 1720 !013
1940 READ O :: FOR I=1 TO O
:: READ D$(I):: NEXT I :: FO
R I=1 TO O !156
1950 FOR D=X TO Y :: IF POS(
A$(D),D$(I),1)<1 THEN 1980 !
197
1960 C=POS(A$(D)," ",1)+1 ::
 B$=SEG$(A$(D),C,LEN(A$(D)))
:: C$=B$&"." :: IF D$(I)<>B$
 AND D$(I)<>C$ THEN 1980 !03
3
1970 S=(VAL(SEG$(A$(D),1,C-1
))*1E6/V AND-1)/10 :: D$(I)=
RPT$(" ",13-LEN(STR$(S)))&ST
R$(S)&" "&D$(I):: CALL SOUND
(1,550,0):: PRINT D$(I):: GO
TO 1950 !104
1980 NEXT D :: NEXT I :: RES
TORE :: GOTO 1720 !224
1990 CALL SOUND(100,310,0)::
 GOTO 180 !255
2000 PRINT "Counts sentences
 containing":" a word in pos
ition,":" referenced to the
start":" or end of the sente
nce.": : : :: W$="    |  |
|       |" !179
2010 PRINT "from start";RPT$
("=",16);">": :"  1   2  3
  4       5":W$:"THIS IS AN
EXAMPLE SENTENCE.":W$:" -4
-3 -2   -1       0": : :: PR
INT "<";RPT$("=",18);"from e
nd": : : : :!150
2020 ON ERROR 2020 !244
2030 C,O=0 :: PRINT "Word, P
osition in Sentence": :" (Ex
ample:  IS,2 or IS,-3)": :TA
B(13);:: INPUT "":W$,A :: PR
INT TAB(4);"#" :: IF A<>0 TH
EN 2040 ELSE W$=W$&"." !186
2040 IF A<0 THEN 2070 !020
2050 IF A$(X+A-1)<>W$ THEN 2
070 !253
2060 C=C+1 :: PRINT USING "#
### ":X+A-1;:: FOR B=X TO A+
X-1 :: PRINT A$(B);" ";:: NE
XT B :: PRINT !040
2070 FOR I=X TO Y :: IF POS(
A$(I),".",1)=0 THEN 2110 !04
0
2080 O=O+1 :: IF A$(A+I)<>W$
 THEN 2110 !243
2090 B=MIN(I,A+I):: IF A<=0
THEN 2100 ELSE B=B+1 !132
2100 D=MAX(I,A+I):: C=C+1 ::
 PRINT USING "#### ":A+I;::
FOR B=B TO D :: PRINT A$(B);
" ";:: NEXT B :: PRINT !140
2110 NEXT I :: PRINT : :TAB(
6);C;"Counted,": :TAB(7);STR
$((C/O*1000 AND-1)/10);"% of
";O;"Sentences.": : :: INPUT
 M$:W$ :: CALL HCHAR(21,1,32
,84):: IF W$="" THEN 2000 EL
SE 180 !093
2120 DISPLAY AT(12,1):" type
 'SIZE' for bytes free": :TA
B(6);"'CON' to continue" !12
7
2130 BREAK !142
2140 GOTO 180 !003
2150 B=100 :: P=LEN(STR$(N))
*2+2 !062
2160 RESTORE :: IF N/B<26 TH
EN 2170 ELSE B=N/25 AND-1 !0
21
2170 W$=" = Cumulative " ::
PRINT A$(0): : :TAB(7);CHR$(
138);W$;"Sum":TAB(7);CHR$(13
7);W$;"Ave":TAB(7);CHR$(136)
;" = Overlapping": : : :!154
2180 PRINT N;F$: : :: READ O
 :: FOR F=1 TO O :: READ B$
:: C$=B$&"." :: A,Z=0 :: CAL
L KEY(0,K,S):: IF K=81 THEN
2310 !174
2190 FOR Y=B TO N STEP B ::
C=0 :: X=Y-B+1 :: N$=STR$(X)
&STR$(-Y):: FOR I=X TO Y ::
IF A$(I)=B$ OR A$(I)=C$ THEN
 C=C+1 !214
2200 NEXT I :: A=A+1 :: Z=Z+
C :: D$(A)=RPT$(" ",P-LEN(N$
))&N$&CHR$(31)&STR$(C):: NEX
T Y :: Z=Z/A !089
2210 PRINT : :B$;TAB(20);CHR
$(127):: D,V=0 :: FOR Y=1 TO
 A :: PRINT D$(Y):: C=VAL(SE
G$(D$(Y),POS(D$(Y),CHR$(31),
1)+1,9)):: I=C-Z :: S,V=V+I
:: X=V/Y :: G=I^2 :: D=D+G !
232
2220 X=X+22 AND-1 :: S=S+22
AND-1 :: CALL HCHAR(23,22,12
7):: IF S>0 AND S<32 THEN 22
40 !086
2230 IF S<1 THEN S=1 ELSE S=
32 !012
2240 IF X>0 AND X<33 THEN 22
60 !236
2250 IF X<1 THEN X=1 ELSE X=
32 !027
2260 CALL HCHAR(23,S,138)::
CALL HCHAR(23,X,137):: IF S=
X THEN CALL HCHAR(23,X,136)!
076
2270 CALL KEY(0,K,S):: IF K=
81 THEN 2310 !133
2280 IF K=82 AND Y>18 THEN 2
210 !235
2290 IF S<>0 THEN 2270 !174
2300 NEXT Y :: PRINT "*Mean=
";STR$(INT(Z*10+.5)/10);TAB(
20);CHR$(127):"*Sdev=";STR$(
INT(SQR(D/(A-1))*1E4+.5)/1E4
)!136
2310 NEXT F :: X=1 :: Y=N ::
 Z=8 :: PRINT : :M$&" or a n
umber  to reset block size:
";:: INPUT "":W$ :: CALL HCH
AR(20,1,32,128):: IF LEN(W$)
=0 THEN 2160 !057
2320 K=ASC(W$):: IF K<48 OR
K>57 THEN 1480 !084
2330 B=VAL(W$):: CALL CLEAR
:: GOTO 2160 !094
2340 I=POS(W$,"-",1):: X=VAL
(SEG$(W$,1,I-1)):: Y=VAL(SEG
$(W$,I+1,LEN(W$))):: RETURN
!242
```

THE ART OF ASSEMBLY — PART 32

Commonly used instructions

By BRUCE HARRISON

Last month we went back to fundamentals, to give those just starting in assembly a first lesson in the language. Today we continue with the primitive operations called "instructions." We'll cover some of the most commonly used ones, and where possible relate them to instructions in BASIC.

The instruction set built into the TI99/4A is a very rich one. Many instructions are provided, so that the microprocessor can perform a wide range of operations. This chip has lots of work to do in following the instructions we give it. It takes each instruction apart bit by bit to determine what's to be done and what needs to be fetched from memory, placed back into memory, and so on. We could start with almost anything, so we might as well start with the MOVE instructions.

A MOVING EXPERIENCE

One of the most commonly used instructions in our programs is the MOV (move) instruction. It's provided in two forms. MOV means we will move a word (16 bits) from one place to another. MOVB means (as you might guess) that we'll move a byte (8 bits) from one place to another. Each time we use either form of the move instruction, we must provide two operands. These operands may be labels, indicating a memory location, or they may be registers, or even combinations of both labels and registers.

Let's start with a simple case that can be related to an operation done in BASIC. Suppose we have a label for a word of data called FLAG1, and we want that word moved into a word used as a variable called VAR1. The instruction in source code would look like this:

```
MOV @FLAG1,@VAR1
```

This is equivalent to an operation in BASIC such as:

```
VAR1 = FLAG1
```

In both cases, the value of the variable FLAG1 is unchanged after the operation, but the value of VAR1 is set equal to whatever was the value of FLAG1. Notice that, in assembly, the variable that's being changed in value is after the comma, while in BASIC, it's before the equal sign. In the parlance of assembly, the operands are in the order SOURCE,DESTINATION. The destination operand gets modified, while the source operand remains the same. In this first case, both operands were memory words identified by labels. Move operations can also be performed using the Workspace Registers. MOV R4,R5 makes R5 contain whatever number was in R4. This kind of move can temporarily "stash" the contents of a register before some operation that will destroy that register's current value. After an operation that changed the value in R4, we could get back its original value by MOV R5,R4 if we'd stashed R4 in R5 before that.

Move operations can also work in either direction between registers and variables. Thus we could put the value currently in VAR1 into R4 by MOV @VAR1,R4. Conversely, we can move the contents of a register into a labeled memory location like our FLAG1 by MOV R5,@FLAG1.

The MOVB instruction works exactly the same, except that when registers are involved, the MOVB always involves the high order byte of the register. If, for example, R4 contained the number >F3E4, doing a MOVB R4,@SAVBYT would move only the eight bits containing >F3 into the location labeled SAVBYT.

INTEGER MATH INSTRUCTIONS

We all know that one of the computer's most basic purposes is to compute. That is, do mathematical functions. The simplest of these operations are, of course, to add and subtract. The instructions in source code are simply the uppercase letters A and S. Like the move instructions, each of these needs two operands, a source and a destination. These can be either registers or labels used as variables, or combinations of the two. Let's start with the simplest possible example, to add the values in two registers. That looks like this:

```
A  R2,R  add the value in register 2 to that in R1
```

As in the MOV case, the source operand remains unchanged, while the destination is changed. Let's say that, before this instruction R2 contained 6 and R1 contained 4. After the operation, R2 will still contain 6, but R1 will contain 10. The same can be done with labels used as variables, as for example the labels PLUS and COUNT. In assembly, we could write:

```
A  @PLUS,@COUNT  add plus to count
```

This is the equivalent of the BASIC instruction COUNT = COUNT + PLUS. The result is that COUNT contains the sum of its previous value and the value at PLUS, while PLUS retains its previous contents.

Subtract, with the opcode S, works in a similar fashion, with results going to the destination operand, and the source operand remaining unchanged.

THE TOUGHER STUFF

In grade school, we learned that the operations of multiplying and dividing numbers were much tougher than adding and subtracting. In BASIC, these operations are just as easy as the others. In assembly, we're back to grade school, and these operations get tough again.

For openers, we can't use anything but Registers for the destination operand in MPY and DIV operations. (The reason for this will soon be clear.) The source operand may be either a register or a label used as a variable. In either case, the MPY and DIV operations affect two adjacent registers, not just one. This is done because the results of either MPY or DIV need two registers to contain the outcome. Let's say, for example, that R3 contains a number (let's say 3) that we want to multiply by 5. We could load another register with 5, then perform the operation, like this:

```
LI  R2,5   place the number 5 in register 2
MPY  R2,R3  multiply register 3 by register 2
```

The result of this operation will be a double-word number in registers R3 and R4. This has to be done because the result of a

(See Page 18)

THE ART OF ASSEMBLY—

(Continued from Page 17)

multiply can be a number too large to fit in one register. In this case, R3 would contain 0 after the multiply, while R4 would contain 15. If R3 contained a number like 32000 in the previous example, then the result would be 160,000. That's too big a number for one register, so it's placed in the pair of registers R3 and R4, with the lower order part in R4, and the higher order part in R3.

Divide operations start with a pair of registers to contain the number that's to be divided. The instruction DIV R1,R3 would divide the double word quantity in R3-R4 by the value in R1. Thus, if the original quantity we wanted to divide were only a one-word value, we'd place that in R4 and clear R3, so that the value to be divided would be correctly represented in the two registers R3-R4.

The results of a divide will appear in the register pair used as the destination, but as two separate numbers. The first register in the pair will contain the quotient, while the second contains the remainder. Let's take a simple example. Assume that R3 contains 0, R4 contains 14, and R1 contains 3. If we then perform DIV R1,R3, the results in R3-R4 will be that R3 contains 4, which is the largest number of times 3 goes into 14, while R4 will contain 2, which is the remainder. Think of it like this:

```
 04
3/14
 -12
  2
```

That's the operation in simple arithmetic, in which we take the trial quotient 4, multiply by the divisor, then subtract that product from the dividend to find the remainder. So long as the remainder is less than the divisor, we are finished with this stage. That's how the result shows up for a DIV operation, with the quotient in the first register and the remainder in the second.

COMPARE and JUMP INSTRUCTIONS

The jump instructions are related in some respects to the BASIC GOTO instruction, but most of them actually perform an IF-THEN kind of operation, and then a GOTO operation. In many cases these "conditional jumps" are used in conjunction with the compare instruction, so we'll treat them together.

Let's start with a simple situation, in which we want to see whether a register contains a number greater than some limit, and to continue the program somewhere else if that's so. In such a case, we could write:

```
C   R5,@LIMIT   compare R5 to the variable LIMIT
JGT  BIG        if greater, jump to label BIG
(else continue program here)
BIG  CLR  R5        clear register 5
```

For the moment, we'll ignore the fact that one of the operands for compare is a register, and state what would be an equivalent in BASIC:

```
300 IF REG5 > LIMIT THEN 500
310 (program executes if REG5 is equal to or less than LIMIT)
...
500 REG5=0
```

Here, the program lines starting at 310 would execute if the variable REG5 were less than or equal to the variable LIMIT, but if variable REG5 were greater than LIMIT, then line 500 would be executed after the IF-THEN.

The compare instruction (C) needs two operands, and these can be registers, labels, or a combination of the two. After a compare, conditional jumps can be made on any of several results of the comparison. JEQ means Jump if Equal, JLT means Jump if Less Than, and, of course, JGT means Jump if Greater Than. Other conditionals exist, such as JNE, for Jump if Not Equal. For some operations, we can use the comparisons on a logical basis rather than an arithmetic basis. Instructions such as JH and JL work that way. To understand the difference, let's take an example, in which we compare R3, which contains >8000, and R4, which contains >0005. If we use C R3,R5 followed by JGT, the jump will not happen, because >8000 is treated as the negative number -32768, and that's less than 5. If we used the logical equivalent JH, the jump will occur, since >8000 is logically higher than >0005.

In many cases, the conditional jumps can be used without the compare instruction. This is so because many operations affect the bits in the status register, so the "compare" is not necessary. For example, we can perform the equivalent of a FOR-NEXT loop this way:

```
LI  R4,4  load register 4 with the number 4
DOIT  (perform some operation)
DEC  R4   decrement the count in R4
NE  DOIT  if not zero, repeat.
(else perform next operation)
```

This works because when we DEC or INC or MOV something, there is an implied comparison to zero. Thus in the above case, each time through the loop we DEC R4, and the result is compared to zero by the microprocessor, so that a conditional jump after the DEC will behave as if a comparison between the register and zero had been performed. In the above case, the operations between label DOIT and the JNE instruction will be performed four times, until the DEC results in R4 becoming zero.

There are two other forms for the compare instruction. CB means compare just one byte, instead of the word that's compared by the C instruction. CI means Compare Immediate, and this special form must have a register as its first operand and an immediate value as its second operand.

The final case for jumps is the unconditional jump, given simply by the instruction JMP. That's like the unconditional GOTO you're familiar with in BASIC. The range of jumps has limits, but that's beyond the scope of the present discussion. (For cases beyond the range of jumps, there's the unconditional B or Branch instruction, which we'll cover in another lesson.)

In our next installment, we'll cover some more instructions, and will try to give our readers some idea how to create real programs with this powerful language.

West Penn 99ers group gets new mailing address

New mailing address for the West Penn 99ers is c/o Mickey Cendrowski, RD1, Box 133, Russellton, PA 15076.

TUTORIAL

Using CALL MAGNIFY

By LUCIE DORAIS

The following article appeared first in the newsletter of the Ottawa TI99/4A User Group.

Lets be serious with a utility that I turned into a real tutorial about that difficult sprite statement, CALL MAGNIFY.

A regular character, made up of 8x8 pixels, is defined by a 16-char. string; but what if you want it bigger — let's say double size? You have the choice of defining a 64-char. string for a 16x16 pixel character, or you can make it a sprite and just MAGNIFY its size. The default for a one-character sprite is CALL MAGNIFY(1), but with CALL MAGNIFY(2), presto, the character is doubled.

But, you might object, with a 64-char. string I can put in more details in my sprite. Right! But if you want a sprite to use all four characters — this is really what the 64-char. definition creates — you must use CALL MAGNIFY(3), and Extended BASIC insists that your first character be divisible by four (100,104, etc.). But Extended BASIC can make this big sprite even bigger, by using CALL MAGNIFY(4), for a sprite of 4x4 characters! Quite effective to make those monsters grow when they approach their poor victim (a few games use this feature).

Here is the program:

MAGNIFY

```
100 ! MAGNIFY / M. MONTMIGNY
, Sherbrooke UG, 1986/modifi
ed by L. DORAIS, Ottawa UG,
Dec. 1993 !149
110 CALL CLEAR :: CALL SCREE
N(2):: CALL CHAR(45,"000000F
F")!189
120 DIM CD$(15):: L$=RPT$("-
",28):: UL$=" "&SEG$(L$,1,16
)!151
130 C$="00007E" :: CALL CHAR
(35,C$&"727242427E"&C$&"4242
72727E"&C$&"4E4E42427E"&C$&"
42424E4E7E")!025
140 T2$=CHR$(132)&CHR$(134):
: B2$=CHR$(133)&CHR$(135)::
T3$=CHR$(136)&CHR$(138):: B3
$=CHR$(137)&CHR$(139)!032
150 DATA 00,03,0C,0F,30,33,3
C,3F,C0,C3,CC,CF,F0,F3,FC,FF
 !016
160 GOTO 170 :: A,C$,CC$,DC,
K,M$,N,S :: CALL KEY :: CALL
 COLOR :: CALL VCHAR :: CALL
 CHARPAT :: CALL MAGNIFY ::
CALL SPRITE :: CALL DELSPRIT
E :: !@P- !064
170 FOR N=0 TO 12 :: CALL CO
LOR(N,12-4*(N>2 AND N<9),1):
: NEXT N !074
180 CALL COLOR(13,3,1,14,3,1
):: CALL MAGNIFY(2):: C$="MA
NIFY" !183
190 FOR N=1 TO LEN(C$):: A=A
SC(SEG$(C$,N,1)):: CALL SPRI
TE(#N,A,2+N,24*N-23,42):: NE
XT N !107
200 DISPLAY AT(20,10)BEEP:"M
ICHEL MONTMIGNY":TAB(10);"CL
UB SHER-TI 1986":TAB(9);"SHE
RBROOKE,QUEBEC" !167
210 FOR N=0 TO 15 :: READ CD
$(N):: NEXT N !087
220 FOR N=1 TO 150 :: CALL K
EY(0,K,S):: IF S<>0 THEN N=2
00 ! force loop exit !049
230 NEXT N :: CALL DELSPRITE
(ALL)!217
240 ! ===== main ===== !234
250 DISPLAY AT(1,8)BEEP ERAS
E ALL:"CALL MAGNIFY": : :"pr
ess  <C>har  <H>ex code" !06
7
260 CALL KEY(3,K,S):: IF S=0
 OR(K<>67 AND K<>72)THEN 260
 :: IF K=72 THEN 290 !010
270 DISPLAY AT(4,1):"enter a
 character" :: CALL KEY(5,K,
S):: ACCEPT AT(4,19)SIZE(-1)
:C$ :: A=ASC(C$)!172
280 CALL CHARPAT(A,C$):: DIS
PLAY AT(4,1):"hexadecimal co
de for "&CHR$(A):: GOTO 300
!228
290 DISPLAY AT(4,1):"enter h
exadecimal code": : :UL$ ::
ACCEPT AT(6,2)SIZE(-16)VALID
ATE(DIGIT,"ABCDEF"):C$ !141
300 C$=C$&RPT$("0",16-LEN(C$
)):: DISPLAY AT(6,2):C$:"" !
 trailing zeroes !084
310 CC$="" :: DC=1 :: GOSUB
470 :: DC=2 :: GOSUB 470 ! m
agnify definition !219
320 DISPLAY AT(8,1):"magnifi
ed hexadecimal code" :: CALL
 CHAR(35,C$,132,CC$)!106
330 FOR N=1 TO 4 :: DISPLAY
AT(N+9,2):SEG$(CC$,16*N-15,1
6)&CHR$(34+N):: NEXT N ! 64-
char string !031
340 DISPLAY AT(5,25):T2$ ::
DISPLAY AT(6,21):CHR$(128)&"
 1 "&B2$&" 2" ! dummy sprite
s mag(1/2) !069
350 CALL CHAR(136,CC$):: DIS
PLAY AT(10,21):T3$ :: DISPLA
Y AT(11,21):B3$ :: DISPLAY A
T(13,21):"3" ! dummy sprite
mag(3) !027
360 CALL MAGNIFY(4):: CALL S
PRITE(#1,136,13,73,209):: DI
SPLAY AT(14,24):"s4" ! real
sprite mag(4) !146
370 ! === menu / table === !
125
380 DISPLAY AT(15,1):L$:"pre
ss <1><2><3><4> magnify":TAB
(7);"<M>odify mag hex code":
TAB(7);"<R>edo   <Q>uit" !20
7
390 DISPLAY AT(19,1):L$:"MAG
NIF  1    2     3     4":"--
---- --- ----- ----- -----":
" chars 1x1  2x2   4x4" !127
400 DISPLAY AT(23,1):"pixels
 8x8 16x16 16x16 32x32":"str
ing 16   16     64     64":!06
6
410 CALL KEY(3,K,S):: IF S=0
 THEN 410 ELSE IF K<49 OR K>
52 THEN 430 !070
420 CALL MAGNIFY(K-48):: DIS
PLAY AT(14,25):STR$(K-48)::
CALL VCHAR(10,3,32,4):: CALL
```

(See Page 20)

CALL MAGNIFY —

(Continued from Page 19)

```
 VCHAR(10,3,124,1-3*(K>50)):
 : GOTO 410 !049
430 IF K=82 THEN CALL DELSPR
ITE(#1):: GOTO 250 ELSE IF K
=81 THEN END ELSE IF K<>77 T
HEN 410 ! redo/quit !144
440 CALL VCHAR(10,3,32,4)::
CC$="" :: FOR N=1 TO 4 :: AC
CEPT AT(9+N,2)SIZE(-16)VALID
ATE(DIGIT,"ABCDEF"):C$ ! mod
ify hex code !135
450 CC$=CC$&C$ :: NEXT N ::
GOTO 350 !199
460 ! ===== magnifying sub =
==== !216
470 FOR N=DC TO LEN(C$)STEP
2 :: A=ASC(SEG$(C$,N,1)):: A
=A-48+7*(A>64)!042
480 CC$=CC$&RPT$(CD$(A),2)::
 NEXT N :: RETURN !143
```

MAGNIFY automatically doubles a 16-character string definition; the routine, explained below, belongs to Michel Montmigny. It is very fast. You can either enter a character from the keyboard — no need to know its hex code, CALL CHARPAT will retrieve it, lines 270-280. Or enter a hex code definition (line 290; line 300 adds the trailing zeroes). Tex will then compute the magnified 64-char. code and display it on screen on four lines, one for each character, that make up the big sprite. A little graphic at the right of each line (chars. 35-38, def. line 130) tells you where each character is located in the sprite (line 330), since their order is rather confusing. The four characters are placed top left, bottom left, then top right, bottom right. So, a big sprite using char. 128 (really 128-131) would read:

```
128     130
129     131
```

Tex will also show you an example of each sprite, with magnifying factors 1 to 4; but since poor Tex can deal with only one magnifying factor at a time, only one sprite is real, the one for MAGNIFY(4) — lines 340-360. A table at the bottom of the screen nicely summarizes all the facts about the four factors: string length, and the number of characters and pixels. See lines 390-400.

At this point you can, if you wish, play with the big (4) sprite (see menu, line 380). Change its magnifying factor <1-4> (line 420, one or four "l" at the left of the CHARDEF string indicate which part of the string is used by Tex) and check that CALL MAGNIFY(1) and CALL MAGNIFY(2) indeed use only the first 16 characters of the definition string, i.e. the upper left character. Or <M>odify the big sprite definition to give you sprite more resolution by putting in more details: lines 440-450 accept your changes on screen, then GOTO line 350 to refresh the display of the sprites magnified 3 and 4.

A word about Michel's routine: when you define a character with 16 chars (hex code), the first character is the hex equivalent of the left block (four pixels, really binary values) of the first row, the second is the code for the right block of that same row, then the third is left code for second pixel row, etc. (See the Extended BASIC manual, page 56.)

What Michel has done is put in a CD$ array the two bytes of hex code that correspond to a double "pixel block" (DATA line 150, READ by line 210). Once Tex has a hexadecimal code to work with (C$, computer as CHARPAT in line 280 or entered by you in line 290), it can build the longer string. It does this in two passes: one for the left blocks (DC=1, line 310), the second for the right ones (DC=2). In the subroutine (lines 470-480), the char. definition is read character by character. In the first pass, the odd-numbered characters are read, 1 through 15. In the second pass the even-numbered characters are read, 2 through 16. The position of the correct byte in the array is calculated by a relational expression. If the character is a digit, ASCII code below 65, the position is ASCII code-48. Otherwise it is a letter (A-F) and the true expression will compute to -1, so the position will be ASCII code-48-7, or A-55 ("B" being ASCII 66, its position will be 66-55=11). See Fig. 1.

Fig. 1

```
150 DATA 00,03,0C,0F,30,33,3C,3F,C0,C3,CC,CF,F0,F3,FC,FF
             0  1  2  3  4  5  6  7  8  9  A  B  C  D  E  F
                                            10 11 12 13 14 15
```

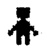
Since doubling a byte really needs four bytes (a 2x2 matrix), two copies of the array value are added to CC$, the magnified hex code string. The chars used for the dummy and real sprites are: 128 for the original C$ hex code; 132-135 are for the

(See Page 21)

First enter the hex code for the little man:

Fig. 2: MAGNIFY(1/2)

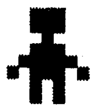
```
383810387CBA2828
```



Tex will magnify the string as:

Fig. 3: MAGNIFY(3/4)

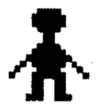
```
0F0F0F0F0303030F0F
3F3FCFCF0C0C0C0C0C
C0C0C0C0000000C0C0
F0F0CCCCC0C0C0C0C0
```



Then play with the magnified hex code as follows (remember, this is just a MAGNIFY tutorial, if not a magnified one. To design good big sprites, many good "spriter" programs are available):

Fig. 4: Modified MAGNIFY(3)

```
070F0F070303070F
1723478F0C0C0C18
80C0C080000080C0
A01088C4C0C0C060
```



When comparing Fig. 2 with Fig. 3, note how the conversion was done by Tex, using the CD$ array in two passes, doubling each CD$ value:

```
                 0F0F 0F0F 0303 0F0F    3F3F CFCF 0C0C 0C0C
pass 1 (CD=1)      3    3    1    3       7    B    2    2
                 C0C0 C0C0 0000 C0C0    F0F0 CCCC C0C0 C0C0
pass 2 (CD=2)      8    8    0    8       C    A    8    8
```

CALL MAGNIFY —

(Continued from Page 20)

magnified two times dummy, and 136-139 for the magnified three times dummy and for the real sprite magnified four times.

To save time, the dummy sprites are displayed as strings (lines 340-350), using T2$/T3$ (top chars) and B2$/B3$ (bottom chars) defined in line 140. We distinguish between dummy 2 and dummy 3 because when you modify the magnified hex code, the dummy sprite 2 is not touched. It is the original 16-char. hex code C$ magnified twice.

To start, see the example in Figs. 2-4 on the previous page.

To leave you with something more entertaining than a tutorial, I found this demo and modified it a bit so that it uses all four factors of CALL MAGNIFY (the unknown author is possibly Craig Miller):

SPRITEDEMO

```
100 ! SPRITEDEMO / unknown /
 mod. L. DORAIS, Ottawa UG,
Dec. 1993 !107
110 CALL CLEAR :: CALL SCREE
N(2):: RANDOMIZE :: CD$=RPT$
("F",64)!043
120 GOTO 130 :: A,B,C,EV,X,Z
 :: CALL CHAR :: CALL SPRITE
 :: CALL DELSPRITE :: CALL M
AGNIFY :: CALL MOTION :: CAL
L PATTERN :: !@P- !216
130 CALL MAGNIFY(2):: CALL C
HAR(100,CD$)! === squares/al
ph === !056
140 FOR X=1 TO 26 :: CALL SP
RITE(#X,100,INT(RND*14+3),1,
1,9,12):: NEXT X !232
150 FOR X=1 TO 26 :: CALL PA
TTERN(#X,64+X):: NEXT X !189
160 FOR X=1 TO 26 :: CALL MO
TION(#X,RND*50,RND+30):: NEX
T X !104
170 GOSUB 300 ! === square s
nakes === !004
180 FOR A=1 TO 4 :: EV=(INT(
A/2)=A/2):: CALL MAGNIFY(3-E
V):: Z=1 !200
190 FOR B=1 TO 2 :: C=9INT(RN
D*14)+3 :: FOR X=1 TO 24 ::
CALL SPRITE(#X,100,C,22,124,
17,X*Z)!004
200 NEXT X :: GOSUB 300 :: Z
=-1 :: NEXT B :: NEXT A !246
210 CALL CHAR(100,"1"):: FOR
 Z=1 TO 2 ! === dots === !21
1
220 FOR C=1 TO 4 :: A=1+2*(C
>2):: B=1+2*(C>1 AND C<4)!18
5
230 CALL MAGNIFY(1-(C>2))::
GOSUB 260 :: NEXT C :: NEXT
Z !175
240 CALL DELSPRITE(ALL):: CA
LL CLEAR :: GOTO 130 ! start
 over !198
250 ! === subs === !014
260 FOR X=1 TO 28 :: CALL SP
RITE(#X,100,16,92,124):: NEX
T X :: IF Z>0 THEN 280 !018
270 FOR X=1 TO 28 :: CALL MO
TION(#X,X,-X):: NEXT X !078
280 FOR X=1 TO 28 :: CALL MO
TION(#X,A*X,B*X):: NEXT X !1
49
290 FOR X=1 TO 1200 :: NEXT
X :: RETURN ! long delay !21
6
300 FOR X=1 TO 200 :: NEXT X
 :: CALL DELSPRITE(ALL):: RE
TURN ! short delay !027
```

1994 TI FAIRS

FEBRUARY

Fest-West, Feb. 19-20, Santa Rita Park Inn, Tucson, Arizona. Contact Tom Wills, Fest-West '94 Committee, Southwest 99ers Users Group, P.O. Box 17831, Tucson, AZ 85731 or (602) 886-2460; BJ Mathis, (602) 747-5046; or the Cactus Patch BBS, (602) 290-6277.

MAY

Lima Multi User Group Conference, May 13-14, Ohio State University Lima Campus, Lima, Ohio. Contact Lima Ohio Users Group, P.O. Box 647, Venedocia, OH 45894.

This TI event listing is a permanent feature of MICROpendium. User groups and others planning events for TI/Geneve users may send information for inclusion in this standing column. Send information to MICROpendium Fairs, P.O. Box 1343, Round Rock, TX 78680.

Miller to produce CD-ROM with TI files

Beery Miller has announced that he plans to produce a CD-ROM to contain approximately 5,000 files for the TI99/4A and Geneve 9640 computers.

He says the CD-ROM will be usable only on MS-DOS computers "unless Bud Mills' SCSI DSR will support proper file extension for MS-DOS."

He says the CD-ROM will contain MS-DOS utilities to unarchive files archived with Barry Boone's Archiver and to convert D/V80 files to IBM ASCII files. With the CD-ROM, TI and Geneve files can be added to a BBS on an MS-DOS machine for downloading by TI users.

He is taking orders for the CD-ROM and estimates the cost at $75 each. If he receives a large number of orders the price could be lower, he says. He asks that customers make checks for the CD-ROM separate from other orders. He says he will cash checks only when the CD-ROM goes into production, around March 1 if orders are sufficient.

The CD-ROM size is estimated to be more than 300 megabytes.

For further information, or to order, contact Beery Miller, 9640 News, P.O. Box 752465, Memphis, TN 38175, or phone his BBS, (901) 368-0112.

NEWSBYTES

Geneve terminal program to support high speeds

Tim Tesch and Mike Maksimik are working on a new color terminal emulator for the Geneve 9640 computer.

The new terminal emulator is not ready for release yet, Tesch says, but will eventually be marketed through Cecure Electronics. The price has not yet been determined, but will be "reasonable," Tesch says.

He says the color ANSI terminal emulator will support high speed modems.

"We have it running at 38,400 baud now," he notes.

He says Port will support multiple types of file transfers, including YModem batch and Mass Transfer multiple protocols. It will have a spooler, log and the "other normal amenities."

For further information, contact Cecure Electronics, P.O. Box 132, Muskego, WI 53150, or (414) 679-4343 (voice), (414) 679-3636 (fax) or (414) 422-9669 (BBS).

JSC newsletter folds

The Johnson Space Center TI99/4A Users Group east of Houston, Texas, has discontinued publishing its newsletter, according to Phil Van Nostrand, newsletter editor.

Van Nostrand says the decision came when free printing for the newsletter, previously available, ended. Some brief issues for the group's members only may be printed, he adds.

He says JSC members have also been encouraged to join other stronger groups such as the Houston Users Group and the Milwaukee group, which has invited the group's members to join as associate members.

Jones to edit TImes

Don Jones of the Chicago Users Group is the new editor of the Chicago TImes, the group's newsletter, according to Don Walden, the group's president, who says he was informed of the change by interim editor Mike Maksimik.

Walden says the group apologizes to those who have not yet received their December, January and February issues, and says the group "will catch up" on the mailings.

Spooner to compile TI Internet directory

Eunice Spooner, sponsor of the Oakland Computer Club, a club mainly for students of the Atwood-Tapley Elementary School in Oakland, Maine, is attempting to compile a list of TI users' Internet addresses and share it with others.

She asks TI users willing to be listed to contact her at her Internet address, ebspoone@colby.edu.

Lima has new dates for 1994

Charles Good of the Lima (Ohio) Users Group has brought it to our attention that our listing of 1994 fairs has previously contained May 14-15, the 1993 dates. The Lima Multi User Group Conference for 1994 is scheduled May 13 and 14.

CECURE®
ELECTRONICS INC.

SALE * SALE * SALE * SALE * SALE

128K X 0 LP 80ns MEMORY HITACHI
Quantities of 4 or more, else $19.95
Quantities limited to 8 per customer.
SORRY, NO DEALER SALES.

NEW * NEW * NEW * NEW * NEW * NEW* NEW
$ 1 5 * $15 * $ 1 5
CFORM * C F O R M * CFORM
$ 1 5 * $15 * $ 1 5

FORMAT YOUR HARD DRIVE FROM MDOS
* USE 34 SECTORS PER CYLINDER *
ADD AN ADDITIONAL 6% STORAGE
1.2 MEG TO YOUR 20 MEG HARD DRIVE
2.4 MEG TO YOUR 40 MEG HARD DRIVE

MBP-II™ MBP-II™
HARDWARE/SOFTWARE

NEW * NEW * NEW * NEW * NEW * NEW * NEW

MBPII CARD

BARE BOARD, MANUAL, SOFTWARE	$27
ABOVE-PLUS ALL PARTS-YOU SOLDER	$67
COMPLETED ASSEMBLY	$87
CLOCK CHIP MM58167AN	$ 9
CLOCK CRYSTAL 32.768 KHz	$ 2
ANALOG-DIGITAL CHIP ADC0809CCN	$ 6

VOICE 414-679-4343
FAX 414-679-3736
BBS 414-422-9669

MASTER CARD AND VISA OR
C.O.D. ORDERS USE OUR TOLL FREE
ORDER ONLY PHONE NUMBER:
1-800-959-9640
ORDERS ONLY!

S. 74 W. 17000 Janesville Rd.
P.O. Box 132, Muskego, WI 53150-0132

READER TO READER

❑ Ted A. Stringfellow, 115 Booth Circle, Ocean Springs, MS 39564-8505, responds to Richard C. Arthur Jr. (Reader to Reader, January 1994), who inquired about saving to the hard drive using Telco:

Richard, you didn't mention which version of Telco you were using. The latest version I know if is V.2.3. You can upload, download or log to a hard drive with no problem. I can also load the Telco program from a hard drive with no problem. Personally, I prefer to have the boot path set as "DSK.TELCO."; in this way, I can keep a copy on floppy disk that I can use on any of my TI systems.

Reader to Reader is a column to put TI and Geneve users in contact with other users. Address questions to *Reader to Reader, c/o MICROpendium, P.O. Box 1343, Round Rock, TX 78680.* We encourage those who answer the questions to forward us a copy of the reply to share with readers.

Ask Marilyn

By HAROLD HOYT

Reprinted from the St. Louis Computer Bridge of the St. Louis 99ers)

The Marilyn Vos Savant column in *Parade Magazine* is always interesting. Somewhere, at a reader's request, Marilyn got the total land surface area of the earth: A=57,160,000 square miles and the total population of the earth, P=5,500,000,000. P/A=96+ people per square mile, kind of crowded. What bothers some of us with a little interest in mathematics is the realization that every year the world population is increasing, and our arithmetic disturbs us.

Try the little BASIC program below. The program asks for what percent you expect the population to increase each year. Try a modest 5 percent. Five percent more people enter the earth each year than leave it. Each year, the amount of space allowed per person decreases. The program has 5280x5280 factored in to convert square miles to square feet. Each iteration checks to see if a well behaved (comatose?) person has enough room. Our politically correct (sensitive, cooperative?), Brave New World inhabitant with suitably low aggression, personal virtual reality vision expander, personal oxygen recycler and food supply (Soylent Green? [food and name of science fiction movie]) might feel alarm when his allotted personal land space shrinks below four square feet (2' x 2' is a little tight). No amount of tranquilizers in his/her Soylent Green ration can quite convince our hero that all is well when his space shrinks to one square foot by the year 2251.

Obviously, something else will happen to make the equation stop working first. Traditionally, we have the four horses of the Apocalypse to help us with population problems. There has to be a better way. Run the little program with your own growth projection percent input. All that will change is the year that the disaster occurs.

Try reading Dr. Paul Erlich's classic, *The Population Bomb* or some of his newer works for some more convincing.

```
1 REM SAVE DSK1.POPULATION !
085
100 REM Program by H.C. Hoyt
 Jr. 11-8-93 !226
110 A=5.716E7 !075
120 P=5.5E9 !241
130 YEAR=1993 !145
140 INPUT "Annual Growth Per
cent: ":G !177
150 P=P*(1+G/100)!176
160 YEAR=YEAR+1 !219
170 DENSITY=A*5280*5280/P !2
39
180 PRINT YEAR;.01*INT(100*D
ENSITY+.5):"Year Square Feet
 Per Person" !066
190 CALL SOUND(1E3,110,30)!1
96
200 IF DENSITY>4 THEN 220 !1
82
210 PRINT "We've got a probl
em!" !082
220 IF DENSITY>1 THEN 250 !2
09
230 PRINT "We've got a BIG p
roblem!" !072
240 STOP !152
250 GOTO 150 !229
```

In Line 1, we have the usual file name, fil saver. We use REM rather than ! in case you want to run the program in console BASIC. Line 110 sets the land surface of the earth, A in square miles and 120 sets the current population of the earth, P. Line 130 sets the current year as 1993 and 140 asks you to INPUT the annual growth rate, G.

Line 150 recalculates the population P for the next year and 160 increments the year by 1. The variable DENSITY in line 170 is really misnamed since it is really the inverse DENSITY, since the equation calculates the inverse of DENSITY, the total square feet per person allowed in the current year. Line 180 displays (PRINTs) the results. Line 190 is a dummy CALL SOUND statement designed to slow the program looping down. The first number, 1E3, is the time of the sound, 1000 milliseconds. Thus, each line is displayed for a little longer than one second before scrolling. The value of the pitch (second number) really doesn't matter, since the third number (30) completely shuts the sound off.

Line 200 skips around line 210 as long as our people have more than four square feet of space. With less than four square feet of space, line 210 displays "We've got a problem!" Similarly, 220 and 230 additionally displays "We've got a BIG problem!" and stops the program at line 240 when the space per person reaches one square foot.

An Extended BASIC program to sort columnar text files

The following program was modified by Woody Wilson of the San Diego TI Special Interest Group to sort D/V80 files. It was set up specifically to work with the group's listing of TI software and hardware, but it can be adapted for other uses. It requires a text file with data entered in columnar format.

Example 1

```
94162^^Adapter,^Dual^Joystick^^^^^^3.00
94046^^Book,^Using/Programing^4A^^^2.00
94004^^Cable,^Dual^Cass.^Lead^^^^^^^.50
```

Here is what Wilson wrote:

Using TI-Writer, you can make your own database by typing your information in columnar format. You'll need to write down column number of each field and change the DATA statements in line 140

(See Page 26)

MICRO-REVIEWS

America's First War, Brukinbox, How to Use The Printer's Apprentice and Not Go Insane

By CHARLES GOOD

AMERICA'S FIRST WAR by Brukin Software

When I was growing up in Chicago as a teenager I dearly loved playing Avalon Hill war games. Good bookstores and toy stores today still stock these fantastic board games, including some of the same titles I played in my youth. (I am now 47.) They are simulations of historical or fictional battles. The historical simulations are realistic and well researched, with a finely detailed map for the playing board and playing pieces representing relative combat values (strengths) of the actual military units that took part in the battle.

These Avalon Hill games take a long time to play and are very complicated, with sets of instructions the size of magazines and playing pieces sometimes numbering in the hundreds. One monstrously large game I remember playing was a complete simulation of the European half of World War II. The game board would be set up on a table in my room for many days while my friends and I worked on a single game.

America's First War is a computerized "Avalon Hill-type" game based on the war of 1812. This was a relatively small war, compared to other wars our nation has fought, but is the only war in which foreign troops occupied and did damage to parts of the continental United States. My home town of Chicago, was the scene the westernmost battle of this war, the Fort Dearborn massacre. Indians, encouraged by the British, surrounded a fort at the site of the Michigan Ave. bridge in what is now downtown Chicago. Safe exit passage was offered to the occupants, who all left the fort on August 15, 1812. After walking a mile and a half, 52 settlers and soldiers, almost all the fort's occupants, were killed.

This two-player game is written entirely in Extended BASIC. Game action takes place around and in Lakes Erie, Ontario and Champlain. An on-screen map is displayed. Players take turns entering data into the computer while the other player is not looking. First, each player decides in which cities and towns to place the troops. Specific military units are available for placement with a variety of combat values (military strengths). There are three types of units: regular military, Indians and ships. Each of these types has its own fighting and movement characteristics as defined in the rules. For example, regular military, but not Indians, can be transported across lakes by ships.

Once troops are initially positioned, players take turns building ships or moving one unit at a time to an adjacent town on the computer's map. If this movement causes opposing forces to come in contact (players can't see the enemy displacement, but the computer knows), a battle begins. The computer displays a picture of the opposing forces as they shoot at each other. Shooting continues and casualties reduce the combat value of both sides until one side either retreats or has its combat value reduced to zero. The game takes three years of historical time with each year divided into 10 turns for each side. Players alternate moving one unit and resolving any subsequent battle. Reinforcements to both sides are delivered to specific towns at the end of each year.

Players get points for territory occupied and number of military units still active. At the end of each year, if the point difference between the two sides exceeds a predefined level, the side with the most points wins the war. At the beginning of the game players can select from three levels of difficulty. The most difficult level requires the greatest point spread to achieve victory.

America's First War (AFW) is certainly not a fast action game. It takes a long time to load and play action is slow. However, strategy games of this type don't have to be fast to be enjoyable. Fortunately, you can save any AFW game position and reload the game later for more play. My Avalon Hill games sometimes took weeks to finish, and I had to keep the game board set up for all this time. AFW is not as complicated, graphically detailed, or as exciting as the better Avalon Hill games. It is, however, the best of this type of war game available for the TI, with more detail and better graphics than any of the war strategy games available from Jim Peterson's public domain library. AFW is fairware and comes on a SSSD disk that includes on-disk documentation. Send me a $1 bill and I'll send it to you to try. Better yet, send a disk, paid mailer, and the requested fairware donation directly to Brukin. Minimum registration fee is $5, which gets you notification of the next update. A $10 donation will result in the next update being automatically mailed to you. Brukin Software, 7919 Mitchell Farm Lane, Cincinnati OH 45242.

BRUKINBOX by Brukin Software

This is not your typical arcade action boxing simulation involving two players with joysticks trying to knock each other out in an on-screen boxing ring. Instead, you pick two contestants from a list of real heavyweight boxers and have them fight each other. The computer determines the outcome of each round based on the contestants' historically determined strengths and weaknesses in various categories.

Some of the 51 boxers you can choose are Muhammed Ali, Max Baer, Jack Dempsey, Ingemar Johansson, Joe Louis, Max Schmeling, Ken Norton, Floyd Patterson, George Foreman, etc. Since many of these individuals are not contemporaries and never fought each other, you

(See Page 25)

MICRO-REVIEWS —

(Continued from Page 24)

can play "what if" games, matching two great boxers from different time periods, such as Ali and Schmeling. Each boxer is rated as either poor, good, excellent or superior in each of the following categories: style, control ability, take punch, power, endurance, defense and toughness. Some are noted to be "better against a power boxer" or "better against a strategy boxer." These ratings largely determine the outcome of a fight, but the computer adds an element of chance as well. Several fights between the same two boxers do not always produce the same winner.

Players determine the number of rounds for the match, and at the beginning of each round the preferred strategy of each boxer. Strategies are "Cover Up" (reduce chances of knock down), "Fight Inside" (go for the body), "Dance and Sting" (increase defensive ability), "Just Dance" (good defense but reduced known down ability) and "Go for Knockout" (decrease defense ability). These strategy options are available for only half the rounds.

Once the fight begins the computer does all the work for you. There is no joystick action or player intervention during a fight. The computer provides a text description and optionally also a graphic display of the fight. This information includes type of punch, punch strength, knockdowns, fouls, cuts, clinches, etc., as they happen. If too many cuts or other injuries occur, the ref may call a TKO. At the end of each round you get a report of each fighter's condition (such as "breathing heavily, cut above right eye"), and a point score awarded by each of the three judges.

Brukinbox is not fast. Written in Extended BASIC, it is slow to load and its optional graphic displays of the fight are small and jerky. The whole concept of the game is, however, unique. I know of no other direct comparison of historical boxers available on any other computer. Another first for the TI!

The game is fairware and comes on a SSSD disk with documentation. I'll send it to you to try for a $1 bill. If you send the requested $15 registration direct to Brukin the company will send you the latest version of the game and mail you notification of future updates. Registered users can purchase for a small additional fee data files for actual historical light heavyweight, middleweight, welterweight and lightweight boxers. Brukin's address is listed above.

HOW TO USE THE PRINTER'S APPRENTICE AND NOT GO INSANE!
by Ken Gilliland

The Printer's Apprentice, also known as TPA, is probably the most sophisticated most full featured desktop publishing software ever produced for the 99/4A. It and its companion programs TPA Toolbox and The Geometer's Apprentice are the closest thing for our computer to a CADD program, allowing one-pass printing of multicolumn text in a variety of fonts mixed with complex three-dimensional drawings. TPA is the only 99/4A software I know that produces text that is both right-justified and microproportioned. The usual (TI-Writer) method of right justification is to leave double or triple space gaps between some of the words in a line to move the end of the line over to the right margin. TPA automatically individually microadjusts all the letters of a line left and right a pixel at a time so that gaps between words in a line are all the same, with double-sized gaps at the end of sentences.

In spite of all its quality features, few 99/4A owners have ever used TPA. I personally know of no TPA users except for Ken Gilliland (although I am sure there are others). Because TPA is so full featured it is also slow and complicated to use. It does not give 99/4A users the luxury of an on screen WYSIWYG display, and its documentation is written in the style of the Editor/Assembler manual. All necessary information is in the official TPA docs, but it is difficult to figure out how to use TPA with only this documentation. Ken Gilliland's book solves this problem.

Ken's 91-page book is a total step-by-step guide to using TPA and its companion software TPA Toolbox. Even I, who have no artistic talent and have never used graphic software for the 99/4A to do more than print pictures created by others, can follow Ken's TPA instructions without difficulty. Ken told me he beta tested his book using a friend who had no previous experience with desktop publishing or the 99/4A. I believe him because in three evenings, by following the instructions in Ken's book, I was using TPA to create really neat posters, letterhead stationary and greeting cards of my own design. (I was, however, still using artwork by made by others to adorn these creations.). The book takes you completely through TPA and the TPA Toolbox, and includes chapters on The Geometer's Apprentice and the TPA version for MDOS. Ken says that the MDOS version of TPA is so good that this software alone is adequate justification for owning a Geneve. TPA on the Geneve is much faster than on the 99/4A, and you get a WYSIWYG display. It's too bad I don't own a Geneve.

Each chapter of *How to Use TPA* ... starts out with an introductory paragraph describing what will be covered and a list of ingredients. The list for chapter 2, for example, includes "1 The Printer's Apprentice software disk, 1 blank disk with the TPA files TREASURE and TYPER copied onto it, 1 sheet of printer paper, a pinch of patience." This last comment is an example of Ken's sense of humor. The whole book is chock full of Ken's unique writing style and humor. For example, the introductory paragraph of the TGA chapter ends with, "So together, let's slide off our printer paper and fall in the third dimension and enter the TGA zone...(do—do-do-do, do—-do-do-do, do—-do-do-do...)"

The entire lavishly illustrated book has been created using TPA. Included are side-by-side comparisons of the same text/graphic pages printed using TPA and using Page Pro. Page Pro is the most common desktop publishing software used by 99/4A owners, and Ken uses the similar side-by-side TPA and Page Pro pages to highlight what he considers the superior features of TPA. A companion disk that comes with the book contains many of the pictures and fonts used in the book.

How to Use TPA Without Going Insane! is a must for all current TPA owners. The existence of the book should encourage

(See Page 26)

MICRO-REVIEWS —

(Continued from Page 25)

more TIers to try to acquire TPA, TPA Toolbox, The Geometer's Apprentice and most particularly TPA for MDOS. If you like Page Pro for desktop publishing you might try using Ken's book and the TPA software instead, particularly the Geneve version of TPA. You may like TPA better. Ken's book is available for $19.95 plus $1.50 shipping from Notung Software, 7647 McGroarty St., Tujunga, CA 91042. TPA for MDOS is available from L.L. Conner Enterprise (phone 317-742-8146), Competition Computer (phone 800-471-1600) and Ramcharged Computers (800-669-1214). *Unfortunately*, these dealers report to me that they are out of stock of 99/4A TPA, TGA, and TPA toolbox. I am told that the TPA copyright holder, Mike McCann, no longer manufactures his 99/4A software and has not licensed anyone else to sell the software. What a shame! *(Dee Turner announced his arrangement to distribute TPA and do periodic updates in early 1993. Contact Turner at 450 Skylark Dr., San Bernardino, CA 92405, or (909) 881-8879.—Ed.)*

Send your 99/4A products to me for reviewing in this column: Charles Good, P.O. Box 647, Venedocia, OH 45895 (phone 419-667-3131).

SORTING COLUMNAR TEXT FILES —

(Continued from Page 23)

to reflect those columns. Be sure to leave a number for a non-existing last field. The "K" in the FOR-NEXT loop in line 170 should match the total number of items in the DATA statement. In other words, if you have three columns to sort on, it would read: FOR K=1 to 4, adding one to account for the required nonexistent field.

When you save your data file, I recommend that you use PF with C DSKn.FILENAME to strip out any carriage returns and tabs. If you don't, you may find that you have those unwanted characters printing out near the beginning of your printout.

To demonstrate how easily FLDSRT18 can sort a D/V80 file, I suggest entering the the lines displayed in Example 1. Replace each ^ with a space.

Lines 140 and 170 reflect the correct starting point of the filed columns and total number of fields. When using this program, I suggest you sort only the number of items you need. If you ask for 200 when you have only 10 items, it will do the sort but takes longer.

FLDSRT18

```
100 ! PROGRAM ADAPTED FROM N
ORTHERN N.J.          FEB 89
 PAGE 3 THRU 5 !172
110 ! PROGRAM MODIFIED AND R
ESEQUENCED SEPT 1993 BY W.A.
 WILSON OF SAN DIEGO SCCG. !
045
120 ! MULTIKEY SORT OF STRIN
G DATA SAVED AS STRUCTURED(B
Y COLUMN) D/V80 FILE !076
130 CALL CLEAR !209
140 DATA 1,8,35,40 !007
150 ! DATA REFLECTS THE COLU
MN NUMBER OF EACH FIELD TO B
E SORTED !210
160 DIM S(4),N$(200)! S() IS
 STARTING COL. OF FIELDS, N$
() HOLDS LIST OF NAMES !203
170 FOR K=1 TO 4 :: READ S(K
):: NEXT K ! READ STRUCTURE
OF FIELDS. IN THIS CASE, THE
RE ARE THREE FIELDS PLUS ONE
 NON-EXISTING TO MAKE 4 !002
180 ON ERROR 660 !159
190 INPUT "ENTER NUMBER OF I
TEMS YOU   WISH TO SORT (MAX
=200) ":WW !214
200 LINPUT "WHAT IS THE FILE
 NAME OF THEFILE YOU WISH TO
 SORT?       ":FI$ !028
210 INPUT "IN WHICH DRIVE IS
 THE FILE  LOCATED? ":DR !03
7
220 OPEN #6:"DSK"&STR$(DR)&"
."&FI$,INPUT :: FOR J=1 TO W
W :: LINPUT #6:N$(J)! READ D
ATA OFF D/V80 DISK !245
230 IF EOF(6)THEN 250 ELSE 2
40 !103
240 NEXT J !224
250 CLOSE #6 !156
260 CALL CLEAR :: PRINT "ENT
ER 0 FOR SCREEN         ENT
ER 1 FOR PRINTER        ENT
ER 2 SAVE TO DISK       ENT
ER 9 TO QUIT     0" !072
270 ACCEPT AT(23,21)VALIDATE
("0129")SIZE(-1)BEEP:WP !WP=
WHERE PRINT? #0 AUTOMATICALL
Y GOES TO SCREEN !155
280 FLAG=FLAG+1 :: IF FLAG>1
 THEN J=J-1 !069
290 IF WP=0 THEN 360 ELSE IF
 WP=1 THEN 300 ELSE IF WP=2
THEN 320 ELSE END !249
300 OPEN #WP:"PIO",OUTPUT !0
84
310 GOTO 360 !184
320 CALL CLEAR :: DISPLAY AT
(12,3):"IN WHICH DRIVE WILL
YOUR    DATA SAVE DISK BE? "
 !158
330 INPUT DR !040
340 CALL CLEAR :: LINPUT "WH
AT WILL BE THE FILE NAME? (U
SE DIFFERENT NAMES IF READAN
D SAVE FILES ARE IN SAME  DR
IVE.)":FI$ !161
350 OPEN #2:"DSK"&STR$(DR)&"
."&FI$,UPDATE !082
360 CALL CLEAR :: PRINT "SEL
ECT THE FIELD ON WHICH TOSOR
T 1-2" :: PRINT !072
370 PRINT "1= SCCCG NUMBER"
!076
380 PRINT "2= ITEM Name " !2
39
390 ACCEPT AT(23,14)VALIDATE
("12")SIZE(-1)BEEP:K1 !220
400 PRINT #WP !064
410 ON K1 GOSUB 610,620 !PRI
NT NAME OF FIELD SELECTED !1
68
420 C=S(K1)! C=COLUMN NUMBER
 OF KEY FIELD !089
430 L=S(K1+1)-S(K1)! L=LENGT
H OF FIELD !038
440 N=J :: GOSUB 490 ! SEND
TO SORT !011
450 CALL CLEAR :: FOR J=1 TO
```

(See Page 27)

USER NOTES

Loading My-Word

This comes from Everett L. Henry II, of Old Orchard Beach, Maine. He writes:

Giving credit where credit is due, here is something about My-Word that my daughter Jennifer found.

When loading My-Word (V1.22) from MDOS (1.53H) Menu using GETKEY and EXEC, if you have the cap lock on and use Shift and Key for menu selection, My-Word will load to run with capital letters only, whether or not the Cap Lock key is on once the program is loaded. Also, the Num Lock key is disabled on the numeric keypad.

I found that if this should happen, there is an easy way to return things to normal. All you have to do is press the End key and everything will be as it should.

Yes I know that you need only to press a key when using GETKEY but, as we all know, programs in the TI/Geneve community all have different keypresses to run and sometimes we forget which we are running and when we forget we find something new.

Adding data screens to mileage program

This comes from John Woestman, of Lancaster, Pennsylvania. He writes:

In reference to Bob August's program in the User Notes of the July 1993 issue of MICROpendium entitled "Keeping track of mileage," the program worked fine until I entered my 18th set of data. The program displays up to 17 sets of mileage data on a screen. The 18th to 35th would be on the next screen, and so on.

One problem is that the "K" in line 2060 is not the same "K" as in line 2130. Moreover, the "J" in lines 2080 and 2090 is fine for the first screen, but "J" increases beyond 17 in the next screen and the next screen should start back at row five (as the first screen does).

I was able to modify the program by replacing "K" in line 2130 with "KK" and by changing "J" to "X" in lines 2080 and 2090. It was also necessary to add the following line:

```
2005 X=1
```

I also changed line 2090 by inserting

```
X=X+1 ::
```

immediately before `NEXT J`. In addition, it is necessary in line 2100 to change `L<N` to `L<=N`.

With these modifications more than one screen can be displayed.

Getting more bang from each byte

This comes from Steve Clarks of Rediware Software, Philadelphia, Pennsylvania. He writes:

There are a lot of articles floating around pertaining to memory conservation on the TI. As we all know, the memory limitations challenge us to be as efficient as possible. However, when constructing a database or similar program in Extended BASIC, we quickly realize our limitations, and I assume all Extended BASIC programmers have dreaded the "MEMORY FULL" error when strings are loaded into arrays. I have found a neat solution to this problem, and have used it in several programs. Here is an except from an article I wrote in 1990 describing my discovery:

As I first began to load the H&D files into PRKBASE, I ran out of memory at 60 pages. I arrayed it as R$(13,103), which would have held all my pages. However, with no memory left, 60 was the limit. To assign a null string, such as A$="" takes 10 bytes of memory. A string with a length of one, such as A$="Q" takes 15 bytes. Every additional character takes one more byte. So a string with a length of 10 takes 24 bytes. Not very efficient!

I then wrote subprograms TAKES and PUT. They take a page of my 13 items and put them into a single string with a character 127 between each item. When I ran out of memory at 60 pages, I had 44 bytes free. After PUT and TAKE, I loaded all 90 pages and still had 10,888 bytes free! Bin-

(See Page 28)

SORTING COLUMNAR TEXT FILES —

(Continued from Page 26)

```
 N :: PRINT #WP:N$(J):: IF J
/7=INT(J/7)AND WP=0 THEN GOS
UB 630 ! SEND TO HOLD SCREEN
 FOR READING !005
460 NEXT J :: IF WP=1 OR WP=
2 THEN CLOSE #WP !083
470 GOSUB 630 ! SEND TO HOLD
 SCREEN 'TIL PRESS ENTER !25
0
480 GOTO 260 ! START OVER AG
AIN !097
490 !**SHELL-METZNER SORT**
!093
500 M=N !089
510 M=INT(M/2)!083
520 IF M=0 THEN RETURN !192
530 K=N-M :: J=1 !234
540 I=J !081
550 P=I+M !101
560 IF SEG$(N$(I),C,L)<=SEG$
(N$(P),C,L)THEN 590 !149
570 T$=N$(I):: N$(I)=N$(P)::
 N$(P)=T$ :: I=I-M !189
580 IF I>=1 THEN 550 !230
590 J=J+1 !013
600 IF J<=K THEN 540 ELSE 51
0 !118
610 PRINT #WP:"BY SCCG NUMBE
R" :: RETURN !152
620 PRINT #WP:"BY ITEM NAME"
 :: RETURN !253
630 PRINT :: INPUT "PRESS EN
TER":A$ :: PRINT :: RETURN !
239
640 CLOSE #WP !068
650 END !139
660 ON ERROR 670 :: CLOSE #1
 !194
670 DISPLAY AT(18,3)ERASE AL
L:"ERROR - CHECK DISK OR DRI
VE!" :: CALL A :: ON ERROR 6
80 :: CLOSE #2 !232
680 CALL CLEAR :: RETURN 180
 !088
690 SUB A :: CALL SOUND(200,
523,0,528,0)!119
700 DISPLAY AT(24,1):"PRESS
ANY KEY" :: CALL KEY(3,A,B):
: IF B<>0 THEN SUBEXIT !063
710 DISPLAY AT(24,1):" " ::
```

USER NOTES

(Continued from Page 27)

go! That problem solved. Now I am remembering 90 strings instead of 1170.

The above description shows the value of such subprograms. The soon-to-be-released RBASE, written for Myarc M-BASIC 3.0, is a database that should handle 400 records, with 13 fields in each record, with a maximum of about 120 characters per record. I have used the same subprograms in RBASE and, if you write anything that requires handling many strings, you will need them too.

```
20 I$="~~~~~~~~~~~~~"
30 OPTION BASE 1
40 DIM A$(13)
```

Line 20 represents an empty record.

```
1000 ! COMBINE STRINGS
1010 FOR Y=1 TO 13 :: A$(Y)&
"~" :: NEXT Y
1020 FR$=A$(1)&A$(2)&A$(3)&A
$(4)&A$(5)&A$(6)&A$(7)&A$(8)
&A$(9)&A$(10)&A$(11)&A$(12)&
A$(13)
1030 RETURN
```

Line 1010 adds the ~ separator to the end of each string. Line 1020 combines the array A$() into one string, FR$. FR$ is now ready to save to a record on disk.

```
2000 ! SEPARATE STRINGS
2010 FOR Y=1 TO 13
2020 P=POS(FR$,"~",1)
2030 A$(Y)=SEG$(FR$,1,P-1)
2040 FR$=SEG$(FR$,P+1,LEN(FR
$)-LEN(A$(Y))-1)
2050 NEXT Y
2060 RETURN
```

FR$ is the string retrieved from a specific record on disk. It contains the 13 pieces of data (fields) together with a ~ between each item. This routine takes the input string FR$ and divides it into 13 separate strings (fields). It also removes the ~ separator.

A search routine can be made much faster by taking apart only the strings that are valid candidates for a match. In the example below, we search for the string SS$, which would match a specific field (FN).

```
850 FOR X=1 TO 400
870 INPUT #1,REC X:FR$
880 IF FR$=I$ THEN 910
890 IF POS(FR$,SS$,1)>0 THEN
960
910 NEXT X
920 etc.........
960 GOSUB 2000
970 IF POS(A$(FN),SS$,1)>0 TH
EN 980 ELSE 910
980 etc........(found match)
```

Line 850 says there are 400 records to check. Line 870 inputs the combined string. Line 880 says that if it is an empty record, don't check it. Line 890 says that only if SS$ is contained in FR$, go separate. Line 960 is a GOSUB to the separate string routine. Line 970 checks to see if the specific string matches.

SYSTEM OF THE MONTH

Custom keyboard makes computing more convenient

Glenn Bernasek, of Strongsville, Ohio, is big on customizing his computer system. This month's System of the Month features a remote keyboard, which he built from an old TI and Televideo units. The TI99/4A console is at the right of the PEB (too dark in the photo to be seen) and includes a small 12-volt DC fan over the vent slots for additional cooling. He uses a 13-inch color TV as a monitor. His Panasonic KX-P1180 printer is partially visible at lower right. An unusual touch is the positioning of the Peripheral Expansion Box on its end in tower fashion. He says it works cooler and, of course, it takes up less space on the desktop.

The PEB operates three floppy drives — one 5.25-inch and two 3.25-inch. The PEB also contains a 384K Chicago RAMdisk. His modem of choice is an Identity ID2400-C modem, located on top of the PEB, which he uses with PC-Transfer and Telco to communicate with other TIers.

Use CAL-PRINT to customize calendars

The following program was written by Harry Allston, of Reedley, California.

The program includes instructions. It allows users to place comments on calendars that are produced on dot-matrix printers. Depending on the printer, readers may have to modify the printer commands.

```
1 ! SAVE DSK1.CAL-PRINT !189
2 ! My Program
   (Public Domain) !201
```

(See Page 29)

USER NOTES

(Continued from Page 28)

```
3 ! Files of Harry Allston
    209-638-7523 !213
4 ! Program has preset tabs
    for each day of week. It
    prints in condensed. You
    are limited to 11 char-
    acters per line for !115
5 ! each entry. For more
    than one line, leave the
    calendar in place and
    repeat the program. Ask
    for the same day and !14
0
6 ! your message will be
    printed under the last
    line printed. ENJOY... !
122
95 CALL CLEAR :: CALL BLUE !
228
100 CALL CHAR(96,"0000FF")::
 DISPLAY AT(3,1)ERASE ALL:"1
994 (TI-PRINTSHOP) CALENDAR"
:"``````````````````````````
`" !225
110 DISPLAY AT(5,1):"This pr
ogram  allows  you to": :"ma
ke additions to a calendar":
 :"made by the T.I. --- (IBM
 or": :"APPLE). Place  month
 page in" !098
120 DISPLAY AT(13,1):"printe
r. Align solid line of": :"t
he day to  top  of  printer"
: :"head.  Enter  addition,
then": :"press ->""ENTER""<-
" !121
130 DISPLAY AT(24,1):"PRESS
ANY KEY TO CONTINUE..." !228
140 CALL KEY(0,K,S):: IF S=0
 THEN 140 !208
150 OPEN #1:"PIO",VARIABLE 1
36 :: PRINT #1:CHR$(15);CHR$
(27);"G";!230
160 DISPLAY AT(3,2)ERASE ALL
:"1994 (PRINTSHOP) CALENDAR"
 !130
170 DISPLAY AT(5,2):"Make ch
anges to:": :"  1. SUNDAY":
:"  2. MONDAY": :"  3. TUESD
AY": :"  4. WEDNESDAY": :"
5. THURSDAY": :"  6. FRIDAY"
: :"  7. SATURDAY" !228
180 DISPLAY AT(21,1):"On wha
t day is the entry?" :: ACCE
PT AT(21,26)SIZE(1)VALIDATE(
DIGIT)BEEP:A$ :: IF A$="" TH
EN 180 ELSE A=VAL(A$)!215
190 DISPLAY AT(23,1):"What i
s the entry?" :: ACCEPT AT(2
4,1)SIZE(-11)BEEP:A$ :: IF A
$="" THEN 190 !192
200 ON A GOTO 210,220,230,24
0,250,260,270,280 !058
210 PRINT #1:TAB(14-LEN(A$)/
2);A$ :: GOTO 160 !146
220 PRINT #1:TAB(34-LEN(A$)/
2);A$ :: GOTO 160 !148
230 PRINT #1:TAB(52-LEN(A$)/
2);A$ :: GOTO 160 !148
240 PRINT #1:TAB(70-LEN(A$)/
2);A$ :: GOTO 160 !148
250 PRINT #1:TAB(93-LEN(A$)/
2);A$ :: GOTO 160 !153
260 PRINT #1:TAB(110-LEN(A$)
/2);A$ :: GOTO 160 !192
270 PRINT #1:TAB(128-LEN(A$)
/2);A$ :: GOTO 160 !201
280 PRINT #1:CHR$(27);"@" ::
 CLOSE #1 !147
10300 SUB BLUE !149
10305 CALL SCREEN(5):: FOR C
=0 TO 14 :: CALL COLOR(C,16,
5):: NEXT C :: SUBEND !179
```

(See Page 30)

USER NOTES

Use RAMID for security on RAMdisks

This comes from William H. Berendts, president of the Ozark 99ers User Group, Springfield, Missouri. He writes:

RAMID is a simple 5-line program written to prevent unauthorized entry to your Horizon RAMdisk. Children, grandchildren and others are often drawn to the mystique of a computer. Ordinarily, they cannot do much harm, unless you have your system autoload your RAMdisk on power-up. Although written for use with the Horizon RAMdisk, you may be able to adapt it for use with other RAMdisks as well. Not having access to other RAMdisks prevents me from testing it on others.

The program is named RAMID. Entering the proper ID will result in your RAMdisk, or any disk protected by RAMID, being made available for use.

If your system autoloads and untrained hands — sometimes even trained hands and minds — hit the wrong keys, havoc breaks loose and you can lose everything stored in the RAMdisk's memory.

RAMID might prevent that by requiring anyone firing up the system to enter an identification code before the RAMdisk menu is loaded. An incorrect identification will result in the following message displayed on the screen: "You are not authorized entry to this computer." It is displayed as an endless loop.

For safety's sake, keep an unprotected copy of RAMID on a floppy, stored in your archive file. Load the program into memory and enter your personal identification. This can be anything — a name, pet name, social security number, address, and so on — into line 1, replacing the statement "your code." If you wish to give another user access to your system under his own personal identification, enter that user's personal code into line 1, replacing the statement "second code." If only one code is used, then delete the second half of line 1 `(AC2$="SECOND CODE")` and in line 2, delete `OR Q$=AC2$`. If you intend to use RAMID to protect a floppy disk, enter in line 2 RUN DSK*.LOAD-FILENAME" in place of DELETE "MENU". If you are protecting your RAMdisk, do not replace DELETE "MENU".

After you have saved the personalized program back to your archive floppy, test it by typing RUN and pressing Enter. Enter an incorrect ID. Your RAM menu (or disk) should not load and the access denied message should flash on the screen, along with a sound to alert the system operator. If you attempt to stop the program by pressing FCTN 4 the loop will continue. The only way to stop the program is to turn off the console, or pressing FCTN +.

So far, so good. Now, reload the RAMID program into memory again. This time enter your correct ID. This time your RAMdisk menu should come to the screen, or the disk drive you wish to access will start, depending on the entry you made in line 2.

If you wish to RAMID to prevent access to your Horizon RAMdisk and everything has worked properly, load your RAMdisk Operating System into memory and reconfigure your RAMdisk by editing the ROS in the following manner: First, move the file ID "4 MENU" to the next empty "U" position. Then enter "4 RAMID" into the position formerly occupied by "4 MENU." Save your altered ROS to another disk or under another filename to prevent changing your original ROS.

After changing your ROS, load your personalized RAMID program into memory and save it to your RAMdisk as a protected file by typing the following:

```
SAVE DSK*.RAMID, PROTECTED
```

The program can be loaded and run, however, it cannot be listed. This prevents an unauthorized user from discovering the personal ID in line 1.

When all loaded, run your system as you normally would. If your system loads on power-up, the line requesting your ID should be displayed and you need only to follow through as described above.

If you decide to change your ID, simply load the unprotected RAMID program from your archive disk, change the ID in line 1 and then save it to your RAMdisk as a protected file, using the filename RAMID.

If you wish to protect a specific disk, save RAMID to the disk under the name "LOAD." Obviously, any other load program on the disk would have to be renamed to avoid overwriting. I have found that just changing the name of the original load program by adding a number to it is easy — LOAD-2, for example. Make sure line 2 of RAMID reflects the changed name, i.e. "RUN DSK*.LOAD-2".

There is one drawback to the use of RAMID — if any program being used is written in other than Extended BASIC, the request for an ID will be displayed when you exit the program. If the programs are written in XBASIC, it might be necessary to make the appropriate change to display the main menu without starting over. Generally, using DELETE "MENU" in place of END will get you back to the RAMdisk menu without going through RAMID. If using a disk, replace END with "RUN DSK*.LOAD-2", or whatever filename you have given the original load program. Programs written in other languages will be more problematical, depending on the user's expertise in those languages.

To receive this program and others on a disk from the author, send a disk mailer and 50 cents postage to: William H. Berendts; HCR-67, Box 485; Roach, MO 65787.

```
1 AC1$="YOUR CODE" :: AC2$="
SECOND CODE" !212
2 DISPLAY AT(8,4)ERASE ALL:"
ENTER YOUR ACCESS CODE:" ::
ON BREAK NEXT :: ACCEPT AT(1
2,5):Q$ :: ON BREAK NEXT ::
IF Q$=AC1$ OR Q$=AC2$ THEN D
ELETE "MENU" !129
3 ON BREAK NEXT :: CALL SOUN
D(500,110,0,130,0,196,0):: O
N BREAK NEXT :: CALL CLEAR !
217
4 ON BREAK NEXT :: FOR X=1 T
O 8 :: ON BREAK NEXT :: DISP
LAY AT(10,4):"YOU ARE NOT AU
THORIZED": :TAB(3);"TO ACCES
S THIS COMPUTER!" :: ON BREA
K NEXT :: NEXT X !115
5 CALL SCREEN(2):: ON BREAK
NEXT :: GOTO 5 !170
```

(See Page 31)

USER NOTES

Ballistics simulation

The following program, by Gene Bohot, was published in Bits, Bytes and Pixels, the newsletter of the Lima 99/4A User Group. It appeared in a column written by Andy Freuh. The program simulates the firing of a cannon at a target.

```
10 RANDOMIZE :: CALL CLEAR :
: !BALLISTICS SIMULATION WIT
H SPRITES !162
20 DISPLAY AT(4,2):"SHOW PAT
H (Y/N) Y" :: ACCEPT AT(4,18
)SIZE(-1):@$ !024
30 CALL CLEAR :: L=45 :: VV=
55 !176
40 FOR N=1 TO 14 :: CALL COL
OR(N,2,15):: NEXT N :: RR=IN
T(20*RND)+10 !097
50 FOR A=1 TO 28 :: CALL SPR
ITE(#A,42,2,256,1):: NEXT A
:: CALL SCREEN(6)!164
60 CALL VCHAR(1,3,124,24)::
CALL HCHAR(24,1,95,32):: CAL
L HCHAR(24,RR+2,31)!056
70 DISPLAY AT(4,2):"TARGET="
;RR :: !011
80 DISPLAY AT(1,2):"ANGLE (1
1-86)=>";L :: ACCEPT AT(1,18
)SIZE(-2):L !251
90 DISPLAY AT(2,2):"SPEED (1
0-70)=>":VV :: ACCEPT AT(2,1
8)SIZE(-3):VV !193
100 IF VV>70 THEN 90 ELSE V=
VV/10 !036
110 A=PI*L/180 :: R=V*V*SIN(
2*A):: COSA=COS(A):: TANA=TA
N(A)!220
120 DISPLAY AT(3,2):"RANGE="
;R :: X=0 :: CL=(ABS(RR-R)<=
1)!015
130 FOR XX=0 TO R-.1 STEP R/
28 :: X=X+1 :: Y=(-1/(2*V*V*
COSA^2))*(XX^2)+XX*TANA !073
140 IF @$="Y" THEN J=X ELSE
J=1 !090
150 IF Y<=24 AND Y>0 AND XX<
=30 THEN CALL LOCATE(#J,192-
8*Y,8*XX+12)!001
160 NEXT XX :: CALL SOUND(-1
00,220,0,-7,0):: IF CL THEN
CALL SCREEN(7):: GOTO 30 !24
1
165 CALL KEY(0,K,S):: IF S=0
 THEN 165 !233
170 CALL CLEAR :: GOTO 50 !2
12
```

CLASSIFIEDS

Policy

The cost of classified advertising is 25 cents per word. Classified display (i.e., special formatting or graphics) is $9 per column inch. Classified advertisements must be paid in advance. Classified advertisers may request a category under which they would like their advertisements to appear, but the final placement decision is the responsibility of the publisher.

Classified deadlines will be kept open for as long as practical. For the purpose of classified advertising deadlines, any classified ad received later than the first day of any month cannot be assured of placement in the next edition. We will do our best to include every advertisement that is submitted in the earliest possible edition.

The publisher offers no guarantee that any advertisement will be published in any particular issue. Any damages that result either from errors in copy or for failure to be included in any particular edition will be limited to the amount of the cost of the advertisement itself. The publisher reserves the right to reject any advertisement.

The advertiser may elect to publish the advertisement in subsequent editions at the same charge, payable prior to publication. The deadline for carryover classifieds is the same as for new advertising.

In submitting an ad, please indicate whether you would like a refund if it is not published in the requested edition or whether you would like us to hold it for the next edition. Cancellations and refunds cannot be made after the second day of the month.

Send classified advertising to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680.

FOR SALE FOR SALE

SEVERAL TI SYSTEMS & parts for sale. For price list, write Jim Ferguson, 22 Welland Crescent, St. Albert, Alberta, Canada T8N 3W4. 11/1

Use MICROpendium classifieds to buy and sell your unused equipment

Buy/Sell Used Hardware and Software

National Used Software/Hardware Club has buyers looking for TI products, as well as sellers. More than 170 TI items are listed for sale. The membership fee is only $15/year and entitles you to buy or sell computer items via the NUS/HC database. You will also receive FREE our newsletter (mailed every 8-12 weeks). If you don't buy or sell anything during the term of your membership we will refund your annual fee. For a free, no-obligation infor mation packet, write to NUS/HC, P.O. Box 1343, Round Rock, TX 78680; or call 512-255-1512. Here are some of the items our members want to buy:

Myarc HFDC	Offer
GRAMKracker (80K)	95.00
Super Space II	25.00
Triple Tech Card	75.00
Rave 105 Keyboard Interface	85.00
80-column card	Offer
TI-Artist Plus	15.00

Call our BBS for periodic price and product updates—512-255-1557. Hours are from 5:30 p.m. to 8 a.m. weekdays and from noon Saturday through 8 a.m. Monday weekends. Supports 300-2400 baud.

The ONLY monthly devoted to the TI99/4A

Subscription Fees

❑ 12 issues, USA, **$35** ❑ 12 issues, Mexico, **$40.25**

❑ 12 issues, Canada **$42.50** ❑ 12 issues, other countries surface mail, **$40.00**

❑ 12 issues, other countries, air mail, **$52.00**

Outside U.S., pay via postal or international money order or credit card; personal checks from non-U.S. banks will be returned.

Address Changes

Subscribers who move may have the delivery of their most recent issue(s) delayed unless MICROpendium is notified six weeks in advance of address changes. Please include your old address as it appears on your mailing label when making an address change.

Check each item ordered (or list on separate page) and enter total amount here:__________

Check/MO ❑ MasterCard ❑ VISA ❑ (check one)

Card No. __________

Expiration Date __________
(Minimum credit card order is $9)

Signature __________
(Required on credit card orders.)

No sales tax on magazine subscriptions. Texas residents add 7.75% sales tax on other items, including back issues and disk subscriptions.

Mail to: MICROpendium, P.O. Box 1343, Round Rock, TX 78680

Name __________

Address __________

City __________

State ______ **ZIP** ______

The set of numbers at the top of your mailing label indicates the cover date of the last issue of your subscription.

Disks, Etc.

❑ **Back Issues**, $3.50 each. List issues:__________

No price breaks on sets of back issues. Free shipping USA. Add 30 cents, single issues to Canada/Mexico. Other foreign shipping 50 cents single issue surface, $1.50 airmail. Write for foreign shipping on multiple copies.

OUT OF STOCK: Vols. 1, No. 1-2; Vol. 2, No. 1; Vol. 4, No. 9

❑ **MICROpendium Index** (2 SSSD disks, 1984-1992), Extended BASIC required $6.00

❑ **MICROpendium Index II** (9 SSSD disks — 1 for each year — 1984-1992), XB required $30.00

❑ **MICROpendium Index II with MICROdex 99** (11 SSSD disks), XB required $35.00

❑ **MICROdex 99** (for use with MP Index II, 2 SSSD disks), XB required $10.00

❑ **MICROpendium Index II** annual disks ordered separately (1 disk per year, 1984-1992); each $6.00

MICROdex 99, by Bill Gaskill, is a collection of programs that allow users of MP Index II to modify their index entries, as well as add entries. MICROdex 99 supports many other functions, including file merging, deletion of purged records, record counting and file browsing.

GENEVE DISKS (SSSD unless specified)

❑ **MDOS 1.23F** (req. SSDD or larger, for MBASIC) $4.00

❑ **MDOS 1.50H** (req. for MBASIC) $4.00

❑ **GPL 1.5** $4.00

❑ **Myarc Disk Manager 1.50** $4.00

❑ **Myarc BASIC 3.0** $4.00

❑ **MY-Word V1.21** $4.00

❑ **Menu 80** (specify floppy or hard disk versions(s); includes SETCOLR, SHOWCOLOR, FIND, XUTILS, REMIND $4.00

GENEVE PUBLIC DOMAIN DISKS

These disks consists of public domain programs available from bulletin boards. If ordering DSDD, specify whether Myarc or CorComp.

	SSSD	**DSSD**	**DSDD**
❑ **Series 1**	$9.00	$7.00	$5.00
❑ **Series 2**	$9.00	$7.00	$5.00
❑ **Series 3**	$9.00	$7.00	$5.00
❑ **Series 4**	$9.00	$7.00	$5.00
❑ **Series 5**	$9.00	$7.00	$5.00
❑ **Series 6**	$9.00	$7.00	$5.00

SECOND CLASS

A T EXPIRES 1/95
[illegible]
P.O. BOX 647
VENEDOCIA OH 45894