CONTENTS

Chairman	John Haller
Editor	Margot Comstock Tommervik
Managing Editor	Al Tommervik
Associate Editor	Craig Stinson
Art Director	Kurt A. Wahlner
Contributing Editors	Roger Wagner
	Jim Merritt
	Peter Olivieri
	Jeffrey Mazur
	William Depew
Editorial Assistant	Linda Va Salle
Operations Manager	Mary Sue Rennells
Administrative Assistant	Kimberly Curling
Circulation	David Hunter
	Ron Rennells
Systems	William V. R. Smith
Advertising Sales	Al Tommervik
(213) 980-5074	Margot Tommervik

Cover design by Kurt Wahlner; photo of Wade and Mark Harris by Jim Salmons.
Composition by Photographics, Hollywood, California. Printing by California Offset Printers, Glendale, California.

DEPARTMENTS

PREVIEWS

Announcing August . . . Robots! . . . Home Ec at Louisiana Tech: Apple Required. . . . Exec Southwestern Data Systems . . . Computer Fever at Folsom Prison . . . SoftCard column debuts: Introducing the CP/M Apple . . . and more . . .

ADVERTISERS INDEX

Softalk. Volume 1, Number 11. Copyright © 1981 by Softalk Publishing Inc. All rights reserved. ISSN:0274-9629. *Softalk* is published monthly by Softalk Publishing Inc., 11021 Magnolia Boulevard, North Hollywood, CA 91601. Telephone, (213) 980-5074. Controlled circulation postage paid at Los Angeles, CA. Postmaster: Send address changes to Softalk, 11021 Magnolia Boulevard, North Hollywood, CA 91601.

SUBSCRIPTIONS: Complimentary to all owners of Apple computers in the USA. Multiple or non-Apple-owner subscriptions for one year: one, $10; two through five, $8 each; more than five, $6 each.
BACK ISSUES: $2 through February 1981; $2.50 thereafter. November and December 1980 issues are sold out. January and March 1981 issues are in short supply.
MOVING? Send new address and old to Softalk Circulation, 11021 Magnolia Boulevard, North Hollywood, California 91601; telephone, (213) 980-5099.

CONTEST: PLACE THE FACE

For the past ten months, *Softalk* has been bringing you the faces behind the companies and products. Some of those faces reappear below. But some have never graced *Softalk*'s pages. Use the pictures and clues to identify these ten people.

1. Left: Datagramming is this software honcho's vision for the future of microcomputer software, and he's got the job to make it happen. 2. Right: If your vision of the Apple includes animation and you'd like to paint your micro in bold colors, he's the man to see.

3. Left: His latest software offering will blast you out of the chair, but it's likely that he won't move. 4. Right: Improving the Apple as a graphics and music device is the Utopian vision of this celebrity.

5. Left: All ads lead to this exec when you answer her company's blurbs. 6. Right: If your screen is a machine or a window that's like magic, thank this man.

7. Left: Mr. fiddled with Fortran, until this Ms. got him on line. She's microcomputerdom's bestselling authoress for the Apple. 8. Right: No prisoner to programming, he looks to the networks for inspiration.

9. Left: Mouthpiece for a fruit . . . and on public television, too. 10. Right: When your programs need editing by the line, you'll want to call on this teenager's software.

1. Using the clues, identify each of the above persons. You do not need back issues of *Softalk* to succeed at this contest. Proper reading of the clues should lead you to their identities.
2. Write their names and current company affiliations on the entry form.
3. Mail entry to Softalk Faces, 11021 Magnolia Blvd., North Hollywood, CA 91601, prior to August 15, 1981.
4. Scoring: One point for correctly identifying the person; one point for correctly identifying each person's current company affiliation.
5. Total possible points: 20. Nearest to that score wins. In case of a tie, the Apple random number generator will select the winner.
6. First prize is $100 in merchandise from any advertiser in this issue. A second prize drawing will be held from all entries that score at least one point. Second prize is $30 in merchandise from any advertiser in this issue.

Fill out and mail to: Softalk Faces, 11021 Magnolia Blvd., North Hollywood, CA 91601.

	Name	Affiliation
1.		
2.		
3.		
4.		
5.		
6.		
7.		
8.		
9.		
10.		

If I win, I'd like: ____________
If I'm second: ____________
Name: ____________
Address: ____________
City, State, Zip: ____________
My autograph: ____________
My local retailer is ____________

LOTSA CONTEST WINNERS

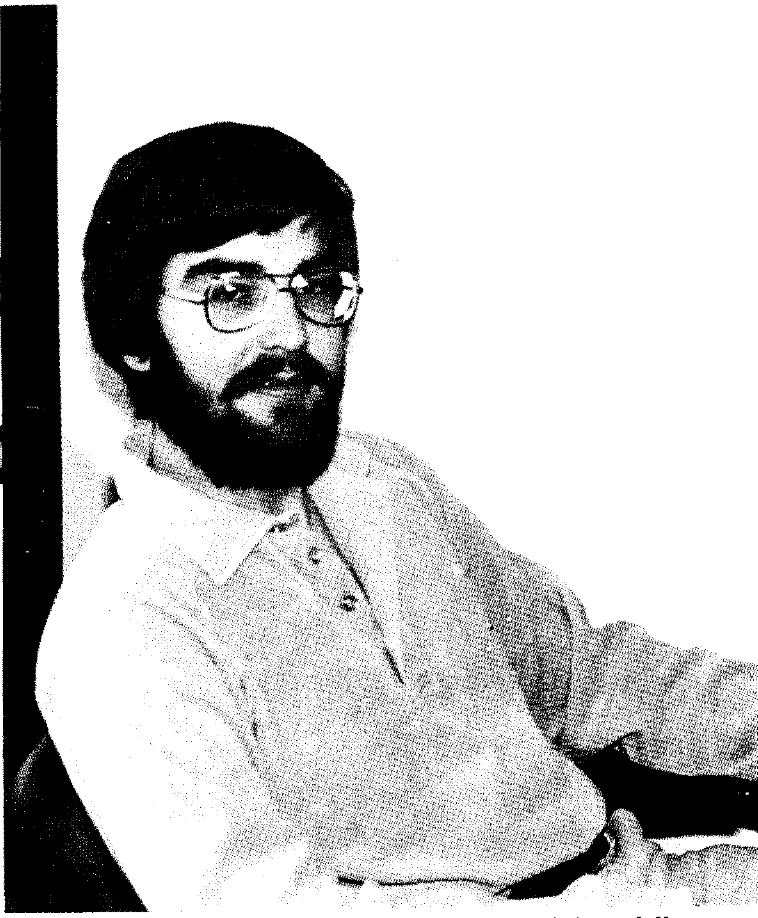
Computer Camp. Kids whose essays didn't win a campership to Computer Camp can take out some of their disappointment by shooting at appleoids. Al Remmers of California Pacific, sponsor of one of the camperships, was so impressed by the competing essays that he's sending a disk of Tom Luhrs's *Appleoids/Chipout* to every entrant.

Todd Kimball of Danville, California, won the *Softalk* Campership. Kimball is twelve. *Softalk*'s staff found his essay best to combine his own purposes at

GOTO 65

Exec Strategic Simulations

BY ALLAN TOMMERVIK


Page 4: Joel Billings, SSI's president. Page 5, clockwise from top left: SSI founder John Lyon, operations manager Susan Billings, marketing manager Jeff Garaventa, and programmer Paul Murray.

Such a contradictory breed are war gamers that they're reminiscent of the old poem about the frog:

"The froggy, him am a queer bird/him ain't got no tail almost hardly/him jumps when he runs/and sits down when he jumps/the froggy, him am a queer bird."

For the most part, war gamers camouflage their interest behind mild demeanors and thoughtful appearances—hardly as macho and bellicose as might be expected. A war games tournament bears great resemblance to a chess tournament, except that the players might seem a trifle unbalanced.

This penchant for slight dementia is most notable during heated debates about the rules—occurrences never found at a chess tournament.

By their very complex nature, war games are not susceptible to clear, concise rule books. And when war gamers congregate, there's bound to be discussion about whose interpretations are correct. Such discussions are generally unsolvable among the participants, necessitating an umpire with the authority to make binding arbitration decisions.

Another trait of war gamers, one more closely akin to their chess-playing cousins, is infinite patience. Players sit for hours, surrounded by their maps, charts, and playing pieces, merely contemplating a move. One move sequence having been completed, they cut card decks, spin spinners, and/or roll dice to determine the outcome of their strategy. As likely as not, the outcome will be a standoff, necessitating a repeat of the entire procedure until one player breaks off the engagement or a conclusive winner is determined.

Not so many years ago, one of the bright young lights in war gaming circles was Joel Billings, then a student at Claremont Men's College. Billings was one of the true talents at the art, sometimes entering as many as three divisions of a war gaming tournament simultaneously. This is roughly akin to playing a couple of dozen simultaneous chess matches against near masters.

Billings's finest hour came when he won one division of a tournament, placed second in another division, and reached the semifinals in yet a third division.

When Billings graduated from Claremont with the bachelor's degree in economics in the spring of 1979, he had an autumn date with the University of Chicago graduate school of business to be preceded by summer employment with Amdahl, a computer mainframe manufacturer in northern California.

He also had his continuing interest in war gaming and a new curiosity about the personal computers that were then just beginning to make their mark.

He perceived that computerizing war games could add to their enjoyment by providing an impersonal umpire, in the form of the computer, who would know beyond a question of a doubt what the rules were.

Also, the computer could carry out both players' moves simultaneously, thus making unnecessary the bogus simulations in which moves are executed in turns, with no chance for the opposing player to react until the result of the first player's move has been determined.

Billings started scouting the possibilities for computerized war gaming and was referred to two programmers at IBM's San Jose facility who were not war game fanatics. Their enthusiasm for the project—initially high—waned considerably when they learned that Billings might have in mind a full-time, full-line software publishing company. They had no objections to some part-time efforts, but they couldn't see the computer war game industry as having the bright future of an IBM.

War gamers are nothing if not thorough, so by that time Billings had developed other avenues of finding programmers who also were war gamers. He distributed questionnaires in stores that featured war games, asking for indications of interest in computerized war gaming and searching for programming talent.

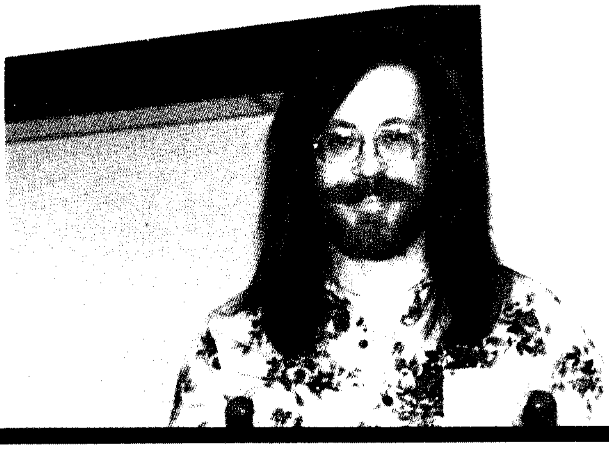
One of the respondents to that questionnaire was John Lyon, who was to become a partner with Billings at the formation of Strategic Simulations.

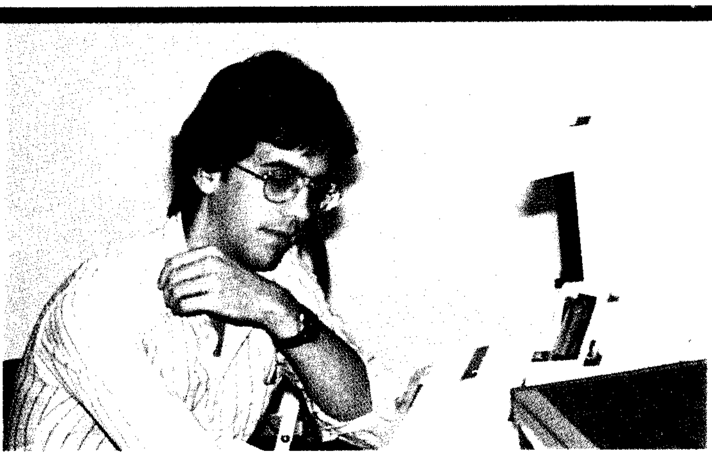
Lyon's background was as checkered as Billings's was pristine. He had dropped out of graduate school—coincidentally, it was the University of Chicago's graduate school of business—several years before.

In the course of doing a friend a favor, he got involved in computer programming. He had given his friend a ride to an employment agency in Chicago and, while his friend applied for a job, Lyon filled out an application form himself. He was sent on an interview for a novice programming position and was hired.

He subsequently went to work for Control Data Corporation, where he spent nearly ten years working at telecommunications systems applications.

At the end of that stint, Lyon was pro-





Softalk photos

grammed out. His disenchantment was so great he swore he'd never program again. He studied gems, but found that jewelry stores were only looking for salespersons.

The result was a return to programming in a totally different environment—he began providing software for Atari pinball machines. Later he went to National Cash Register as an applications programmer.

After leaving Control Data, Lyon had become interested in painting ancient miniatures. As a war gamer, he chose ancient over other period miniatures because there was a consistent set of rules followed nationally for the use of ancient miniatures in gaming and no such consistent set of rules existed for use of miniatures of other periods.

It was during a search for new miniatures at the Game Table in Campbell, California, that he saw a mimeographed flyer soliciting interest in computer war games. At the bottom was a request for interested programmers to call a phone number.

In Lyon's words, "I thought to myself, 'This is what opportunity looks like when it comes knocking.' " He called the number and reached Billings.

Also responding to that questionnaire was Ed Williger. Lyon became author of SSI's first product, *Computer Bismarck.* Williger authored the second product, *Computer Ambush.*

But it wasn't all that simple. In the beginning, Billings and Lyon hadn't even determined which computer might be best for the type of product they were interested in producing. So, while additional market studies were being made, development of *Bismarck* started on a borrowed North Star computer.

At the West Coast war gamers convention, Pacificon, Billings conducted a survey—with a fifty-dollar door prize drawing as an incentive for participation—to determine how many war gamers already had computers and which kind they had.

That survey was followed by telephone interviews with a representative sampling of the initial respondents.

Apple and TRS-80 were approximately even in ownership by war gamers, with all other personal computers together having only a minuscule share of that segment of the marketplace.

Meanwhile, summer was coming to an end and Billings had to bite the bullet. He had to decide whether to cast his lot with SSI or proceed with his plans for graduate school.

SSI won the nod. By that time Lyon had programmed a relatively simple fox and hounds game on the borrowed North Star; that achievement convinced Billings that if they formed SSI, the company would be able to deliver product. The decision was made to convert the North Star code for the Apple, and business was set up in Billings's apartment.

While Lyon set about the code conversion and the continuing programming effort, Billings bent his talents to marketing. One of the results was a breakthrough in computer software packaging.

Bismarck was the first software marketed in a bookshelf size box, emulating its board game predecessors. Although the box is now slimmer, SSI has continued its policy of bright, four-color boxes as its means of packaging.

Bismarck also broke new ground in providing plastic movement charts, maps, grease pencils, and other manual game accoutrements.

These innovations stemmed less from a conscious effort to distinguish the product from other software games than from Billings's familiarity with board war games, which usually contain maps and charts and are boxed.

As with other embryonic companies in the microcomputer software industry, SSI's genesis gives lie to the old term *cottage industry.* It's been an *apartment* industry. Like the other software publishers with similar roots, packaging materials, documentation, and disks soon overran all available living space.

In addition, an Apple and a TRS-80 were set up in Billings's bedroom. When Lyon would finish his regular job at NCR, he'd go to the apartment and program well into the night—the beeps and whirrs and clicks of the computer providing a lullaby for Billings.

Lyon went full-time with SSI in November 1979, as he and Billings strove to wrap up their first product.

The original management team was

completed with the December addition of Susan Billings, formerly a hospital administrator in Santa Barbara.

Bismarck was introduced in late January 1980 and, by early March, it was clear that either Billings or SSI would have to vacate the premises. Naturally, it was SSI that did the moving—the first of two moves to larger quarters within a year's time.

For a company that originally set out to do war games for the Apple, SSI has developed a remarkably diverse software portfolio. *Computer Quarterback* was the first of what may become a full line of sports simulation games for the computer. A second product in that line, a computerized baseball simulation, was being play tested as this article was being written and should be on the market by the time you're reading it.

Cartels and Cutthroats marked SSI's first venture into business simulation games. It was written by Dan Bunten, also the author of *Computer Quarterback.*

The company's product flow stuttered at the beginning but now has fallen into a pattern that calls for a new product each month.

Ambush didn't follow *Bismarck* until June 1980 and it was September before *Quarterback* and *Computer Napoleonics* reached the market.

Computer Conflict and *Computer Air Combat* followed in November, but it was February 1981 before *Warp Factor* made its appearance. *Apocalypse* and *Cartels* were released in March and *Torpedo Fire* hit the market in April.

In the works at the present time is SSI's first battle game involving elements of fantasy games such as magic and mythical monsters. Fueled by Lyon's interest in ancient miniatures, it's a simulation with mythical elements such as Homer might have written.

Computer Bismarck remains the all-time bestseller of the company, but a war game with a twist—*Warp Factor*—is bidding for that honor. Written by Paul Murray, it's a war game of the future, not of the past, and it's currently one of the hottest pieces of software on the market.

The company is constantly working to upgrade the product line from a programming standpoint. There are plans for some of the older products to be converted to assembly language to enable the programs to run faster, although *Ambush* has so far defeated the best efforts of three programmers who attempted it. SSI has also developed its own disk operating system, which speeds access to the disk considerably, an important consideration because the games require constant disk accessing.

Billings is hesitant to draw the parallel because of the obvious competitive aspects, but his goal is essentially to make SSI the publisher of a complete spectrum of strategy games for the computer, just as Avalon Hill has been successful in publishing a wide line of board strategy games.

SSI's success was noted in the 1980 game survey conducted by *The Space Gamer,* the magazine for fantasy gamers. *Bismarck* and *Ambush* rated first and second as the most popular computer war games.

SSI was among the most recognizable of the computer game manufacturers and were rated in the top ten of all gamemakers in the same survey.

Billings believes that the company's image as a war game publisher has prevented some authors from offering other types of strategy games to SSI. He hopes that the sports and business simulations will convince authors to give SSI a shot at their efforts.

Still dedicated war gamers, Billings and Lyon are looking forward to this year's Pacificon in San Mateo; it will be doubling as the national Origins convention of war gaming. They'll be exhibiting instead of playing, but it's the first time the national convention has been held on the West Coast and there's a certain regional pride in the recognition that gives to western war gamers.

If, at that convention, someone has come up with a board game simulation of software publishing, bet the mortgage on Joel Billings to win. He's a proven war gamer who's also a proven publisher. ❑■

OPEN DISCUSSION

Apple on Drugs

We have five Apple computers here at Salem Hospital. The one in the pharmacy is being used extensively as we develop our own software. We showed some of these programs at the New England Council of Hospital Pharmacists spring seminar at Nashua, New Hampshire, in May.

Two of our pharmacists have also purchased Apple computers for use at home.

We're interested in talking with any other pharmacists who have developed programs for hospital pharmacy.

F. Michael Lyons, Director of Pharmacy, Salem Hospital, Salem, Massachusetts

One-Key Printing

I read "Homespun Hopes" in Open Discussion (*Softalk,* May 1981), and I have found one of those little hidden functions that Doug Chang mentioned. To the best of my knowledge, it is not in the Applesoft manual.

It was actually a programming error that I intentionally made. I typed in a program that would, theoretically, never end. So, just for the heck of it, I entered END as the last line of my program. I thought I would be funny, so I put a question mark after the END statement, that is, END?. I didn't think anything of it until I listed the program. Instead of being END?, it was END PRINT. I then subsituted a question mark for PRINT while programming, and, after hitting return and running the program, it worked! To make a long story short, in Applesoft, you may substitute a question mark for PRINT while programming. It is a lot faster! Please try it.

Charles M. Haire, Las Vegas, NV

Mr. Haire's method of discovery is a far more delightful one, but the use of the question mark as an abbreviation for PRINT can also be discovered on page 70 of the Applesoft Basic Programming Reference Manual.

Enough To Turn My Green Eyes Blue

When I bought *Apple Post* software twenty months ago, I had long golden hair to my waist. I now have straight gray hair and am almost bald from fighting "error encountered" problems with the program.

I called Apple, CA, a dozen times. Then I called Apple, TX. I hired two software experts. I parked on the local Apple Computer rep's doorstep. All this was in addition to bugging the local dealers.

No one . . . and I mean no one could tell me what caused the "error encountered" or how to get around it. I spent more time retyping clobbered disks than running programs.

I finally went to CAA for a mailing program and redid my ten thousand plus names.

Most purchasers of *Apple Post* are probably dumb bunnies like me. We get no help from the experts.

The solution is simple:

```
CATALOG
DELETE ZIP
DELETE SOUNDEX
```

Go back into *Apple Post* and these two modules (where the error has occurred) will set up new files without errors.

It also works faster starting from scratch. I would advise deleting those two files before any update.

I don't know a RAM from a ROM or a Control-D from a GOTO, but I know when I have problems.

Claudine Moffatt, Manchester, MD

Softalk *has not encountered this problem and has not tested Ms. Moffatt's rather radical fix. If you recognize these problems and wish to attempt the fix, it's safest to save these programs on a separate disk rather than delete them entirely.*

tsiugniL eht htiw tnemugrA nA

If you look at page 69 of the May 1981 issue, you will see an ad from Synergistic Software for their *Linguist* package.

Line three of the screen is supposed to be Hebrew, Well, folks, someone really goofed! The letters are Hebrew, all right, but you've (or they've) got them going the wrong way.

In Hebrew, we read from right to left (just like APL, hmmm). But in the ad, the screen shows them backward. Not a very good ad for a package which supposedly teaches linguistics.

Also, the proper translation is "Who speaks Hebrew?" not "Do you speak Hebrew?"

Perhaps you should tell your advertiser (or better—proofread the ads you accept).

I don't think I'll buy that package.

L. P. Lewis, White Plains, NY

The program itself does indeed print Hebrew to the screen and on the printer from right to left, thanks to the facility of the Apple. Typesetters are often not so *well equipped. (See Marketalk Reviews.)*

Incidentally, "Do you speak Hebrew?" seems a reasonable idiomatic translation, just as On parle francais *("one speaks French") and* Se habla espanol *("Spanish speaks itself") are properly taken to mean "French spoken here" and "Spanish spoken here."*

Minot To Reason Why . . .

I teach computer programming to high school juniors and seniors here in Minot. We have twelve Apple II Plus units with disk drives and language systems. Our semester course is based on Pascal, touching on Fortran the last four weeks. Our advanced class works with Pascal, Fortran, Pilot, and machine language.

I have enclosed copies of two programs that show how to use special output features in Pascal. I hope you can use these hints.

The first program defines a character variable T as a blank. It would also have been declared as a constant. This makes the T;10 is the Writeln similar in appearance to the T Format control in Fortran.

The variable LF is assigned the linefeed character. I get tired of seeing students' programs, programs in periodicals, and books cluttered with Writelns. Some of them have twenty in a row. Using the LF variable is similar to the / Format control in Fortran.

The VT variable tabs straight down the screen in the middle of a Writeln. I would normally use a name like MC instead of Movecursor in my own programs. This function can easily be added to the System.Library.

```
PROGRAM DEMO:
  VAR VT,LF,T:CHAR;
  PR:INTERACTIVE;
BEGIN
  RESET (PR,'PRINTER:');
  VT := CHR(10);
  LF := CHR(13);
  T := ' ';
  WRITELN ('LINE ONE',T:10,'TAB DEMO'
  LF,'LINEFEED DEMO',
  VT,'VERTICAL TAB DEMO');
      END.
```

```
PROGRAM DEMO:
  VAR VT,HT: INTEGER;
FUNCTION MOVECURSOR(X,Y:INTEGER): CHAR;
  BEGIN
    GOTOXY(X,Y);
    MOVECURSOR := ' ';
     END;
BEGIN
  HT := 20;
  VT := 20;
  PAGE(OUTPUT);
  GOTOXY(10,10);
WRITELN('HERE WE ARE',MOVECURSOR(HT,VT),'NOW
HERE WE ARE');
     END.
```

```
PROGRAM DEMO;
  BEGIN
    WRITELN('THIS IS A BUNCH OF');
    WRITELN('WRITELN STATEMENTS');
    WRITELN('USED TO WRITE ON THE');
    WRITELN('SCREEN');
      END.
```

```
PROGRAM DEMO;
  VAR LF:CHAR;
  BEGIN
    LF := CHR(13);
    WRITELN('THIS IS A BUNCH OF',
      LF,'WRITELN STATEMENTS',
      LF,'USED TO WRITE ON THE',
      LF,'SCREEN');
      END.
```

Could you tell me why the T Format control was left out of the Apple Fortran?

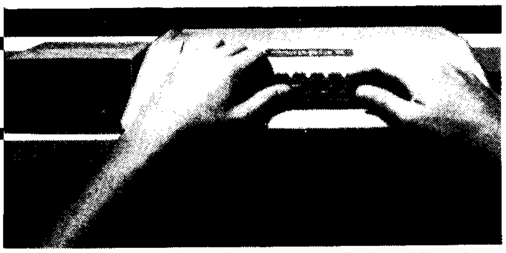
Craig Nansen, Magic City Campus, Minot High School, Minot, ND

Adventuresome Youth

I am a fifteen year old who's in the process of learning machine language and dabbling in Pascal. I have known Applesoft and Basic for quite a while now and enjoy programming very much. I am also a software nut; I especially like adventures, and your magazine has helped me greatly. This leads up to my problem. I have played almost every adventure on the market today, from *Zork* and *Adventure* to *Prisoner* and *Mystery House*, and not one has kept me busy for more than four days. So please, if you or your readers have information on new, *hard!!* adventures, please let me know. Thank you.

Gary Auerbach, Hartsdale, NY

TRADETALK



□ What a robin redbreast is to spring, such is Organizing (the capital oh intended) to an industry's coming of age. So it was no small thing when a few software publishers and a few programmers got together, separately, at the West Coast Computer Faire to lay the foundation for professional interest organizations. The former, temporarily dubbed the Software Publishers' Association, dedicated itself to establishing industry standards for hardware and software. Just since the Faire, it has expanded its scope to include software authors, and it's considering organizing under the modified name of **Association of Software Producers.** In June, about fifty participants and observers representing software interests throughout the United States met at Boston's Applefest '81 to exchange ideas and concerns important to producers. The purpose of the organization is twofold, according to steering committee chairman **Mitch Kapor (Micro Finance Systems,** Cambridge, MA): it will serve an advocacy function, addressing itself to issues such as tax treatment of software royalties and legal protection of software rights; and it will serve as a forum for discussion and coordination among producers. The committee will also propose membership requirements and dues structures, Kapor said. He stresses that the term *producer* subsumes authors as well as publishers and invites both to send suggestions, solutions, and questions to him at Micro Finance, 180 Franklin Street, Cambridge, MA 02139. Other members of the new organization's steering committee are: **Jennifer Bodine, Riverbank Software** (Denton, MD); **Alan Boyd, Microsoft** (Bellevue, WA); **Stan Goldberg, Micro Lab** (Highland Park, IL); **David Lindbergh, Lindbergh Systems** (Boston, MA); **A. Richard Miller, Miller Micro Systems** (Natick, MA); **Evan Scharf, Information Unlimited Software** (Berkeley, CA); **Barney Stone, Stoneware** (San Rafael, CA); and **Ken Williams, On-Line Systems** (Coarsegold, CA).

□ **Vernon Publishing Services** (Vernon CT), a company specializing in sales support items for the life insurance industry, has branched into software publishing by acquiring 50 percent ownership of the **L.P.A. Tech Corporation** (Springtown, PA). L.P.A. adapted the *Capital Need Analysis* and *Financial Need Analysis* computer programs. Vernon created them in noncomputer form.

□ Through a cooperative advertising and promotional program from **Automated Simulations** (Mountain View, CA), dealers and distributors can receive credit toward purchase of promotional goods—T-shirts, catalogs, and newsletters, for example—based on retail value of each order they submit. Credits may be used at time of purchase or accumulated.

□ **Steve Peterson** has taken the reins as president of **Edu-Ware Services** (Canoga Park, CA), following his graduation from UCLA. Peterson will continue to serve as chief financial officer. **Sherwin Steffin** becomes chairman of the board and director of research and development for all products.

□ **Information Unlimited Software** (Berkeley, CA) announces **Chris Hegarty** as chairman of the board. Lecturer and consultant to more than four hundred Fortune 500 corporations, Hegarty is also author of *How To Manage Your Boss* (Rawson, Wade: 1980). IUS has expanded its facilities with a new products development office.

□ **Peachtree Software** (Atlanta, GA) has acquired **Small Business Applications** (Houston, TX), maker of *Magic Wand* word processor for CP/M Apples. Peachtree's modified, more powerful *Magic Wand* will be distributed through its three-hundred-plus dealers worldwide.

□ **John Williams** has returned to the helm as general manager of **On-Line Systems** (Coarsegold, CA). He's also in charge of advertising. **John Harris,** formerly of Gamma Scientific, has come onboard as On-Line's first Atari 800 programmer. Incidentally, **Ken Williams** reports that several hundred people responded to his *Softalk* ad for programmers to program in the wilds of Coarsegold. What does it all mean?

□ **Apple Computer** (Cupertino, CA) heard the cry of dealers and customers who've had to look all the way to Cupertino for technical help. Apple has moved its technical support into the field, creating technical centers in Sunnyvale and Irvine, California; Boston, Massachusetts; Charlotte, North Carolina; Carrollton, Texas; and Toronto, Ontario, Canada. . . . New Apples are certified by the Federal Communications Commission as complying with commission standards for electromagnetic interference. As a result, Apples cost more and weigh more, and the disk drive interface cord is short and battleship gray instead of conveniently long and bright with Apple's rainbow colors—but your neighbors can presumably glut themselves on "As the World Turns" without static. Rebels may note that rainbow-colored interface cords (not a product of Apple), in various lengths, are available separately in many computer and electronic stores.

□ Someone, maybe you, will win a 32K Apple system with two disk drives and a twelve-inch Sanyo monitor in **Inmac's** summer sweepstakes, designed to introduce the micro community to Inmac's new lifetime-guaranteed diskettes. Entry blanks are available in the summer Inmac catalog, which you can get by writing to Inmac, 2465 Augustine Drive, Santa Clara, CA 95051. Contest ends August 10.

THE BASIC Solution

By Wm. V. R. Smith

This month the Basic Solution uses two of the subroutines in your library to create a stand-alone program titled Video Tape Calculation (VTC).

VTC uses the dollar formatter and the screen read subroutines. The dollar formatter is the heart of a numeral input routine introduced in this program.

The screen read routine is used to provide a video "paper" tape of the past calculation. You may wish to delete this subroutine to speed up the execution time.

VTC uses the Reverse Polish Notation implemented in most programmable calculations. A value is entered and the return key is pressed, the second value is entered and then the function to be performed is pressed, +,—,*,/. The * and / functions provide immediate answers. The equal sign, =, must be pressed to show the result of + or —. The C key will clear VTC at anytime.

Your comments are always appreciated. Send your suggestions to *Softalk* Basic Solutions, 11021 Magnolia Boulevard, North Hollywood, CA 91601.

```
10   REM ***********************************
20   REM **
30   REM **              SOFTALK
40   REM **       VIDEO TAPE CALCULATOR
50   REM **
60   REM
70   REM **
80   REM ***********************************
90   HOME : POKE 35,21
100    VTAB 22: PRINT "VIDEO TAPE CALCULATOR
       COMMANDS :"
110    VTAB 23: PRINT " + - * / = C:CLEAR
       RETURN:+"
120    VTAB 1: HOME
130    DIM C$(12)
140    HOME
150  X = 1
160  DP = 35: VTAB 20: HTAB DP - 1 + DX
170    GET C$(X)
180  C = ASC (C$(X) )
190    IF C > AND C < 58 THEN 700
200    IF C = 3 THEN TEXT : END
210    IF C < > 8 THEN 250
220    IF DX > 0 THEN DX = DX - 1
230  X = X - 1: IF X = 0 THEN X = 1
240  C$(X) = "": GOTO 710
250    IF C < > 46 THEN 280
260  DX = DX + 1: IF DX > 1 THEN DX = 1: GOTO
       710
270    GOTO 700
280    REM *** RETURN OR " + "
290    IF C = 13 THEN 310
300    IF C < > 43 THEN 370
310    REM
320    IF X = 1 THEN 350
330  SY = SY + VAL (C$)
340    HTAB DP + 2: PRINT "+";: GOSUB 760
350    GOSUB 920
360    GOTO 750
370    REM *** " - " ***
380    IF C < > 45 THEN 450
390    IF X = 1 THEN 170
400    HTAB DP + 2: PRINT"-";
410  SY = SY - VAL (C$)
420    GOSUB 760
430    GOSUB 920
440    GOTO 750
450    REM *** " = " ***
460    IF C < > 61 THEN 520
470    GOSUB 760:V = SY: VTAB 19: HTAB 30:
         PRINT "----------": VTAB 20:
         GOSUB 960
480    PRINT "T";
490    GOSUB 760
500    GOSUB 920
510    GOTO 750
520    IF C < > 67 THEN 550
530    REM *** " C " ***
540  SY = 0: GOTO 470
550    REM *** " * " ***
560    IF C < > 42 THEN 610
570  SY = SY * VAL (C$)
580    HTAB DP + 2: PRINT "*";
590    GOSUB 760
600    GOTO 470
610    REM *** " / " ***
620    IF C < > 47 THEN 670
630  SY = SY / VAL (C$)
640    HTAB DP + 2: PRINT "/";
650    GOSUB 760
660    GOTO 470
670    REM
680    REM ADD IN YOUR OWN FUNCTIONS
690    GOTO 170
700  X = X + 1
710  C$ = C$(1) + C$(2) + C$(3) +C$(4) + C$(5) +
       C$(6) + C$(7) + C$(8) + C$(9) + C$(10) +
       C$(11) + C$(12)
720  V = VAL (C$)
730    HTAB 20: VTAB 20: CALL - 868
740    GOSUB 960
750    GOTO 160
760    REM
770    HTAB 20: VTAB 1
780    PRINT " ";
790    GOSUB 830
800    CALL - 912
810    HTAB 1: VTAB 20: PRINT T$
820    RETURN
830    REM **** SCREEN READ ROUTINE ****
840  T$ = ""
850  BA = PEEK (40) + PEEK (41) * 256
860    FOR X = PEEK (36) TO 39
870  C = PEEK (BA + X)
880  T$ = T$ + CHR$ (C)
890    NEXT X
900    RETURN
910    GET A$: PRINT ASC (A$): END
920    FOR X = 1 TO 12:C$(X) = "": NEXT X
930  DX = 0:C$ = ""
940  X = 1
950    RETURN
960    REM *******************************
970    REM **      DOLLAR FORMATTER       **
980  V1$ = "":NF = 0
990    IF V < 0 THEN V = ABS (V):NF = 1
1000 V1 = INT (V):V$ = STR$ (V1)
1010 V = ((V - V1) + 1.001) * 100
1020 L = LEN (V$)
1030   IF L < 4 THEN 1070
1040 V1$ = "," + RIGHT$ (V$,3) + V1$
1050 V$ = LEFT$ (V$,L - 3)
1060   GOTO 1020
1070 V$ = V$ + V1$ + "." + MID$ ( STR$ (V),2,2)
1080   IF NF = 1 THEN V$ = "-" + V$
1090 L = LEN (V$)
1100   HTAB (DP - L + 2)
1110   PRINT V$;
1120   RETURN
```

VENTURES WITH VISICALC

BY CRAIG STINSON

If you're an ardent *VisiCalc* user, you've probably heard by now: there are some fancy new kids on the block. Personal Software has finally upgraded its celebrated number cruncher to DOS 3.3, and, more or less at the same time, the company has also released a small armada of related business software. The new line-up includes *VisiPlot, VisiTrend, VisiDex,* and *VisiTerm,* as well as the venerable *VisiCalc* itself.

VisiPlot takes data from *VisiCalc*—or any other source—and converts it to graphic display; *VisiTrend* manipulates the data in various ways and performs statistical analysis upon it. *VisiDex* is a cross-referencing electronic index card program, and *VisiTerm* is a communications package.

In this month's column we will review the first four new products. A review of *VisiTerm* will follow next month.

Besides compatibility with DOS 3.3, there are two other major areas of improvement in the new *VisiCalc.* First, there's a new storage format that eases the transfer of data from *VisiCalc* to other programs (or from other sources to *VisiCalc*); second, *VisiCalc* now includes a number of conditional and logical operations that were missing before.

What's the DIF? The new filing option is called Data Interchange Format (DIF). Here's how it works. When you save via DIF you set your cursor at the upper left corner and specify the lower right corner of the area you want filed. You don't have to save the whole work sheet.

The program sends your data to disk either row by row or column by column, according to your choice. Which you choose will depend on what you want to do with your data later on. Each column or row, depending on which way you save, will be treated as a separate series of data points for the purpose of plotting or analysis by *VisiPlot* or *VisiTrend.*

The DIF procedure stores only sequences of numbers and labels. Formulas, formats, and the original *VisiCalc* coordinates all get left behind. Numbers are filed in full precision, although, if you don't want all those decimal places, you can round off before filing.

DIF is intended to facilitate transfer of data from one application to another. For temporary storage of a *VisiCalc* work sheet in progress, you probably wouldn't want to use this option, because reloading the DIF file would require re-creating all those formulas and formats.

There are some situations, however, in which it might be useful to reload a DIF file onto a *VisiCalc* work sheet. For one thing, when you load, you get the same column-versus-row choice. So, if you have any reason to, you can reorganize data from columns into rows, or vice versa, using DIF.

More importantly, DIF data can be brought back to any part of the *VisiCalc* work sheet. So, for example, if you had several people in the field gathering and storing data via *VisiCalc,* you could consolidate their input on a master work sheet, and the master could both summate the individual input and display it separately.

Until now, Apple *VisiCalc* work sheets could be overlaid, but any data you wanted to juxtapose in the overlay had to come from distinct coordinates to begin with—which required hassle upfront.

The DIF feature permits the integration not only of one function with another—*VisiCalculating* with *VisiPlotting,* for example—but also of one *VisiCalc* application with another.

New Conditions. The other major improvement in the updated *VisiCalc* is the inclusion of conditional operators. There's now a function called @CHOOSE that works like an On . . . Goto statement in Applesoft. @CHOOSE (E7,10,-8.98,F2) first looks at coordinate E7; if that location holds the number 1, the function returns the value 10; if E7 holds a 4, the function yields the value held by F2, and so on. The function only works, of course, if the possible values at E7 are consecutive positive integers.

Other kinds of conditional branching can be brought about through the use of Boolean operators. Now you can plug in a formula like A1/A2 < A3 at a given coordinate, say A4. The formula will yield *true, false, error,* or *NA* (the last case if you're holding A1, A2, or A3 open for data to come). Then, at another coordinate, you can apply an if . . . then . . . else type of branch. The function @IF (A4, 100,200) looks for a Boolean value at A4; if that value is true, the function returns 100; if false, 200. The first field has to be a coordinate location holding either true or false; the second and third fields can be anything.

So now you can put extensive alternate scenarios into distant, unrelated sections of a *VisiCalc* work sheet and summon results from different corners using the conditional operators. *VisiCalc*'s range of uses have taken on new dimensions of complexity.

Two other changes are important. First, the new *VisiCalc* will run on either 3.2 or 3.3 systems. If you don't want to Muffin over your old 3.2 data files, you just throw in the Basics disk before booting *VisiCalc.* Second, the program now takes advantage of the extra memory provided by the language system. The *VisiCalc* is larger than the old one, so if you don't have the language system, you'll find yourself with a little less available workspace in RAM. If you do have the language system, you'll come out slightly ahead.

Plotting against VisiCalc. Of the other new products from Personal Software, the most natural companion to and extension of *VisiCalc* is the *VisiPlot/VisiTrend* package. *VisiPlot* is available as either a stand-alone or as part of the *Trend* composite, so we'll consider it first.

The first thing one is apt to notice when booting up *VisiPlot* (or *VisiTrend*) is a comfortably familiar style. Wherever possible, the outward appearance and manner of *VisiCalc* have been preserved. Communication with the user is via a status area, commands are preceded with the slash key, and the program is menu-driven throughout.

Although it will process any numerical data you care to feed it, *VisiPlot* has been designed to use DIF files created by *VisiCalc.* When you transfer a full or partial work sheet from *VisiCalc* to *VisiPlot* by way of DIF, each row or column depending on how you saved the work sheet, becomes a single plottable set of data points. The entire DIF file goes on disk as one unit, with each row or column a subunit. The program uses the term file to designate the overall structure and series for the subunits. Up to sixteen series, or a maximum of 645 data points, may be loaded into memory at one time; if the *VisiCalc* sheet you're saving exceeds those limits, you'll need to break it up.

Applied to DIF data or not, *VisiPlot*'s file and series approach makes sense, because it allows you to load or save a lot of related information all at once. With eight or ten or sixteen data series in memory at the same time, you can overlay several graphs—up to a maximum 150 data points—for quick and easy comparison.

The kinds of graphs *VisiPlot* offers include line charts, area charts, bar graphs, pies, high-low plots, and scatter plots.

The line chart option draws simple graphs with points connected by straight lines. Up to three can be displayed simultaneously, with the data points of each series marked by different symbols and a legend provided below. The program will actually display as many line plots as you want, up to the maximum 150 data points, but it has distinct symbols for only three series.

Bars to the Left, Bars to the Right. The area chart is simply a line graph with the area between the line and the x axis shaded in.

Bar graph bars come in three styles: normal, left, and right; the left and right afford side-by-side comparison of two data series. Bars can be open, shaded, or hatched, allowing easy distinction in side-by-side overlays. You can also combine two bar graphs in an over-and-under fashion, to illustrate, for example, growth in various categories from one year to the next.

The high-low and scatter charts require two data series over a common range. The high-low is useful for something like tracking high and low stock prices per day for a given stock; on this graph, you could also overlay a line chart of the closing prices. The scatter plot puts one series on the y axis and the other on the x axis and marks all points of coincidence; it's useful for demonstrating the presence and manner of correlation between two series of data.

The different kinds of graphs can be combined in various ways—such as a line over a bar graph or a line over an area chart. Using the window option, the hi-res screen can also be split in half—vertically or horizontally—to display two entirely separate minigraphs at the same time.

Titles come in boldface or regular type. The program provides up to three lines of legend and an unlimited number of labels that you can move around on the graph. These movable titles can be displayed either normally or in inverse video.

Aside from the normal editing functions of changing, adding and deleting data, *VisiPlot* can also interpolate any number of values between two existing points. It can even drop in any specified number of points in an arithmetic or geometric series of your own devising.

Additional features include color control of both data display and background and the option to include horizontal and/or vertical grid lines. You can also restrict the range along either axis, making it possible to zoom in on a particular area of your data. If you alter a scale so that some of your points are off the screen, the program politely tells you you've done so and asks for confirmation. Then it displays what it can.

VisiPlot graphs can be printed on the Silentype, Trendcom 200, Paper Tiger 440 and 445, or the graphics-equipped NEC Spinwriter 5510, 5515, 5520, and 5525. Hi-res images can also be saved as such on disk and incorporated into your own programs.

Taking Care of Business. *VisiPlot* is intended for business use. All the various graph formats expect to see dates along the x axis. That's nifty for most business applications. If your data is monthly, the program will even drop in initials for the appropriate months. On the other hand, if you're going to plot pH against milliliters of titrate, you'll find *VisiPlot* a touch clumsy.

The only other caveat worth mentioning is the possibility of overkill. For those who do not require or want all that *VisiPlot* provides, the program might be unnecessarily unwieldy. It's too large to fit into memory all at once, and going from the plotting module to the file-handling/editing module can cause some vexing delay.

The same comment holds for the composite of *VisiTrend* and *VisiPlot*. In this package, there are three separate, large modules—the plotter, the trender, and the file-handler/editor. For some reason, the program does not allow direct movement from plot module to trend module; you have to go by way of the file handler, an expedition that may leave you with hands in lap for a minute or more.

That's a trivial annoyance, however, considering the power of the combined *Trend/Plot* package. The *Trend* menu offers three major areas of functionality, under the headings Analyze, Function, and Xform.

Analyze displays several series of data in tabular format—up to the limits imposed by Apple's forty-column output. It also calculates various statistical indices and performs linear multiple regression analyses on as many as five independent variables.

The statistics available include: number of points, minimum and maximum, mean, variance, standard deviation, and the coefficient of correlation between any two series.

The regression analysis is done by the least squares method. Besides constant and coefficients, the program also calculates the T statistic, R-bar squared statistic, corrected R-bar squared statistic, the standard error for the regression and for the coefficients, the sum of the squared residuals, the F statistic, and the Durbin-Watson statistic. The regression analysis also generates two new series of data that can be displayed by the plot program: a fitted series, which uses the calculated constant and coefficients; and a residual series, which shows the difference between actual and fitted data.

The second main division of the menu, Function, manipulates original data to provide more information: options are moving average, single exponential smoothing, percent change, lag, lead, and a running total. The moving average is calculated by the N + 1 method, although the user can choose to convert the results to the N or centered methods.

Each manipulation generates its own ancillary data series, which can be plotted alongside the original data.

In case the choices under Function still don't give you the angle on your data that you want, Xform allows you to define your own transformation formulas, using up to eighty characters. All Applesoft operators and functions are at your disposal: arithmetic operators, logical operators, comparative operators, trigonometric and logarithmic functions, and miscellaneous functions like INT, RND, and SQR. Among other things, Xform lets you do the equivalent of plotting a second or higher-order function on log paper. Using the EXP function, you can also make higher-order data available for linear multiple regression analysis.

All Hands on Dex. Turning now from complexity to artful simplicity, *VisiDex* is a freeform data storage and retrieval program. The basic unit of storage is a forty-column by twenty-line index card, filed under user-specified keywords or dates. Within this simple, flexible format, *VisiDex* offers myriad search, sort, print, and calendar options for myriad potential uses.

Like *VisiPlot* and *VisiTrend, VisiDex* employs a code structure and status display that will provide fond remembrance of *VisiCalc.* When you've finished creating a data card, for example, you hit /K for keyword and designate one or many words, numbers, or phrases that will become index entries for subsequent retrieval of the card. Up to eighty characters may be used in keywords for any given card. The current date—or any other—can also be used as a keyword, a feature that permits the use of *VisiDex* as an appointment book and as a file cabinet. If you have a clock from Mountain Computer or California Computer Services, *VisiDex* will handle date functions automatically. Otherwise, you just tell it the date when you boot up.

Keywords don't have to be part of the screen's contents. Designating a keyword that doesn't actually appear on the screen could be handy at print time. You could, for example, file a bunch of names and telephone numbers, all under the keyword phone; you could then print all the cards stored under that heading, without having to print the keyword itself with each entry.

Another way to accomplish this would be to put the keyword on the screen in inverse video; the program gives you the option of ignoring inverted material on the printout.

For that matter, you could put the keywords near the bottom of your index cards and tell the program to print only the top five or ten lines. The print routine defaults to twenty-two lines per card, which leaves two line spaces between each, if you fill the cards, and makes for an even three per form on fanfold paper. But of course, you can override the default value.

Other data entry and editing features include five fixed tab stops, flashing display, a stack-oriented line delete and insert that allows you to move whole lines around on the screen, and word wraparound. The program also supports the Dan Paymar lower-case adaptor, but only with the optional shift-key modification. Without the shift-key mod, the Paymar chip is of no avail.

Besides the freeform approach to data storage, *VisiDex* also offers the option of defining and laying out recurrent data fields. You can set up a template card with fields for, say, name, address, phone number, age, income, or whatever. Then, anytime you want to create a card with those fields, you just call up the template—by its own keyword, of course. The predefined fields on the template draw the cursor, saving keystrokes, and if you put them in inverse, they can be included or excluded at print time.

Data retrieval can be accomplished either by way of keywords or by literal portions of screen text. The keywords, of course, bring it back a whole lot quicker, since the program only has to scan an index; when you hit /KG (get keyword), the disk starts to spin even before you type the keyword. Once you've typed it, retrieval is practically immediate.

In case you forget how you've spelled a keyword, there are two kinds of wildcards at your disposal. The asterisk can stand for single characters, the hyphen for any number of characters.

There's also an option to review the entire index of keywords. The program presents them in alphabetic order and shows you how many cards you've stored under each keyword.

Since the program knows how to sort alphabetically, you can, if you like, devote an entire data disk to a single purpose, such as mail or phone list. If all your keywords are surnames, you can ask for a sorted print and have *VisiDex* run off your entire list—or any segment of it—in alphabetic order.

You can search for numbers in screen text using relational or comparative symbols like greater than or less than. For example, an agency that solicits contributions could establish a list of donors and selectively recall those who had contributed within a particular range, and so on. The program will ignore dollar signs, commas, and decimals during a numeric search.

The Dating Game. The calendar features allow you to make dated reminders to yourself. When you use a date as a keyword, the system also asks you to specify a warning period—between zero and fifteen days. Then whenever you boot *VisiDex,* the program starts by reminding you of any current messages. You can also ask *VisiDex* to display the reminder at various regular intervals—weekly, monthly by date, or monthly by weekday.

The system also displays or prints on command the calendar for any month in the twentieth century with brackets around any date for which there's a reminder stored on disk. Or you can get it to display or print a daily, weekly, or monthly schedule of stored reminders.

Unlike *VisiCalc, VisiPlot,* and *VisiTrend, VisiDex* does not use or create DIF files. It does, however, provide the option of printing a text file to disk, in the manner of *VisiCalc*'s /PD command. The text file thus created can, of course, be incorporated into other programs.

VisiCalc is by Dan Bricklin and Bob Frankston; the price is $199.95. *VisiPlot,* as a stand-alone, sells for $179.95; the price with *VisiTrend* is $259.95. Both programs are by Mitchell Kapor. *VisiDex,* by Peter Jennings, retails for $199.95.

All four programs require 48K and a disk drive. *Plot* and *Trend* need Applesoft as well.

Mind Your Business

BY PETER OLIVIERI

This month, we'll look at the stages a typical business goes through in implementing its first computer system, and we'll consider ways to prepare yourself for that first computer application.

All businesses, regardless of size, must keep records, manage operations, and plan for the future. Toward these ends, the computer can provide very useful service. Improved speed and accuracy enhance record-keeping and can reduce clerical costs. Operations may be improved by more effective inventory control, reduced product costs, and better scheduling. Planning can provide better information to managers, enhancing decision making. Historical data can be used to make predictions for the future.

Sounds great! But there's no turnkey business computer system that arrives on site and manages the business for you. Before the computer can help, you have to do your homework. You'll have to do a lot of self-evaluation along with substantial document and data collection. In addition, you must make a commitment to spend a significant amount of time planning each new application. Over the years, this column will provide resources to help you in these efforts.

There are, indeed, stages firms go through in applying the computer. First, the emphasis is on cost-reduction accounting applications. The usual applications include payroll, customer billing, and accounts payable and receivable. When comfortable with these areas, the business often turns to applying the computer in other functional areas. The computer may be used for inventory control, forecasting, general ledger, capital budgeting, and personnel management.

Typically, there comes a time when there are so many new applications that the business calls a halt to any new programs and turns instead to an emphasis on control. Initially, this might just be control over the existing applications. Eventually, various scheduling and purchasing control applications may be instituted.

The final stage occurs when the business turns toward data base applications. At this time, you'd find a system with financial planning models, simulations models, and, perhaps in each, the capability of direct inquiry into the data base itself.

The Impact of the Microcomputer. The microcomputer's impact on these stages has been rapid, because computer programs necessary to accomplish many of these applications are currently available for microcomputers. (This does not necessarily mean that such programs are easily implemented in a business, a common misconception.) Thus, you could actually start at any of the stages. Despite the temptation to start at an advanced point in this typical growth pattern, you are probably better off starting with the more common and easily managed applications, growing into the more complex and demanding ones. Not only does this pattern improve the likelihood of early successes, it also provides the groundwork for you to plan the computer's role in other business areas.

The Most Common Applications. The most common business applications on the computer are billing and sales analysis, accounts receivable and payable, payroll and labor analysis, sales order/transaction processing, general accounting, and inventory control. We'll study these applications in more detail in a future column.

Regardless of the application you select as the first candidate for computer processing, you need to take certain preliminary steps.

First, make a list of what information you need to run your business. How many transactions are there? What type of information is needed? What reports are required? What should these reports look like? What files are needed to produce these reports? What data is needed in each of these files?

The most important point made here is the need for organization. Being careful and thorough will pay off later. Each area you're considering for computerization must be thoroughly analyzed. You'll need to look at current forms, study existing procedures, interview personnel involved, and gather lots of data. This will be true whether you're computerizing an existing application or designing an entirely new one and whether you're writing the programs yourself or buying software.

It's useful to put together a notebook containing the fruits of your labor. The notebook should contain the following sections:

Section 1: A Narrative. This section briefly describes the application. At the least, it should include the objectives of the proposed application.

Section 2: Interviews. Unless you're truly a one-person business, you'll have to talk with others about their current and future information needs. Record the information gathered through these interviews for reference.

Section 3: The Inputs. During the familiarization process, try to gather all the information you can about the input requirements.

What volumes are involved?
What form does the input take?
When is the input received?
How is the input recorded?
From whom is the input received?
Is the input modified or manipulated in any way?
Where is the input filed?
Is there a retention time for the document?
Where do copies go?

If available, samples of any input forms should be collected.

Section 4: The Outputs. Determining the output requirements is one of the more critical steps. Once you know what output is required, it's rather easy to ensure that you have the necessary inputs to produce those outputs. In addition, you'll be more aware of what processing must take place. Questions you should answer include:

What does the output (or report) look like?
Where does the output go?
What is the purpose of each report?
How long does it take to produce the output?
How long is the report kept?
How are errors corrected?

Once again, samples of any existing reports should be collected. If none exist, rough drawings of the proposals should be provided.

Section 5: The Operations. The input and output documents are linked by the processing that transforms one into the other. You'll need to find out as much as you can about these processes.

What is done?
Who does it?
When is it done?
How is it done?
Are there any special formulas?
How much time does it take?
What rules are followed?
What are other people's perceptions of the system?

Section 6: Nature of Files. Every business maintains records. Your investigation of file requirements will help determine your computer system's storage needs. Collect this data:

What is the name of the file?
Where is it located?
How is it stored—in a filing cabinet? On cards?
How old is the oldest record in the file?
How old is the youngest record in the file?
How is the file organized?
Is it alphabetical? By part number?
How many records are there in the file?
How many fields are contained in each record?
How many characters are contained in a record?
How will data be deleted from the file?

Section 7: Resources. What resources will be needed to implement effectively the proposed application?

Will additional people be needed?
What equipment will be required?
Are there any forms that will be needed?
Will the new system impact available space?
What supplies will be required?
What will the financial commitment be?

As you can see, lists can be a great help in preventing you from forgetting to gather a particular piece of data or ask a certain question. These lists aren't exhaustive, but they provide a beginning point from which to organize your computer planning efforts. We'll continue with such check lists in future issues.

Now, if you've been thorough in gathering background information, you can determine whether your Apple system is complete enough to handle the application that interests you. One key factor is the volume of data that must be stored. A diskette or cassette tape can only hold so much. (The fact is, a tape system is inappropriate for any serious business application and should not be considered an option.) If your application has a large amount of data, you may need more than one disk drive. Indeed, your data storage needs may be so great that the application is presently impractical.

In addition, your review of the output needs for your application will have pointed out the requirements for a printer. Should the printer be able to handle special forms? Must it produce multiple copies of a document? What size paper will be required?

It is critical that you have the right hardware for the job. The best software doesn't work quite as well with inadequate hardware, and great hardware can't improve poor software. Don't consider the minimum system for your needs; rather, choose the most appropriate system. Remember that there are other applications you'll be considering. What will their hardware requirements be? Plan for growth.

Let's assume you have the hardware to implement the application you've selected. Your next step should be to check the market for software packages that meet the needs you've described. Do this even if you are a crackerjack programmer and are considering writing all your own software. There'll be plenty for you to do, and there's no sense in reinventing the wheel. See what's available. Find out the costs. Evaluate. Then make your decision.

Unfortunately, this isn't any easy task. There's such a proliferation of microcomputer software that it's very difficult to find out what's available and determine what's right for you. *Softalk* is an excellent source of information. You might also consult catalogs such as *Vanlove's 1981 Apple Software Directory "Master Catalog,"* published by Vital Information (Kansas City, MO) and *The Book of Apple Computer Software 1981*, published by The Book 1981 (Marina Del Rey, CA). There are, however, very few sources for software evaluation, such as user ratings. While this situation will eventually change (we'll be reviewing and comparing packages in this column), the change will be slow. For now, decisions must be made with incomplete information.

A good beginning is to rely on the reputation of the vendors. Call them. The cost of the call is apt to be small compared to the value of the information you'll receive. You'll be surprised at how much information you can get this way. The vendor won't build your system for you; but, if you've done your homework, you're likely to get direct answers to most of your questions. By the way, have your questions ready before you dial that number.

After reviewing what's available, you may decide that it's better to write your own programs or have someone else write them for you. The packages that interested you might be too expensive, or perhaps your needs are very specific and would require too much customizing. In any event, because of your earlier efforts, you'll be prepared to make an informed decision.

There are a few packages that a business should have because they provide such a variety of applications. One of these packages is a data base management system (hereafter called a DBMS). This is a powerful package that talks the user through an application. The DBMS can create your data base, make modifications, and print a variety of useful reports. The specific application for the DBMS is up to the user.

A second package recommended for any business software library is *VisiCalc* or a *VisiCalc*-like program. This type of package is designed to serve a variety of needs; it is quite a valuable resource.

In our next tutorial, we'll look closely at data base management systems and how they can be applied in business. Our software review section will then describe and evaluate some currently popular DBMS packages.

Review Preview. Mind Your Business is receiving for review many of the business application packages available today. We spend a good deal of time with each, reading all the supporting material, familiarizing ourselves with the particular package, and thoroughly testing it. As the number of application packages increases, we'll provide the basis for making product comparisons. This way, buyers can refer to a table comparing all the packages in an area on a variety of characteristics. You can weight the different characteristics as you see fit to reflect what's important to you.

You can participate in the review process by sending *Softalk* your reactions to systems with which you've had experience. This will build a file of user ratings to share with readers.

Product reviews appearing in Mind Your Business will comment on each of the following: What application area is the product best suited for? What equipment is needed to use this product? What is the quality of the documentation? Is it professionally done? Is it easy to read? Comprehensive? Complete? What is the cost of the product? What documentation is provided with the package? Is the product currently available? Can the applications be customized by buyers for their particular businesses? Can the package be expanded or enhanced as the business grows? What support is provided? Is the vendor accessible? Are backups provided? What about updates to the package? Are there companion products that make this package more powerful? What do other users say about this product?

And, of course, we'll add personal comments when appropriate.

The Readers Speak. "I have an Apple II Plus and I also have a business. I originally got the Apple for home use, primarily for entertainment. It would seem, however, that I should be using it in my business. My problem is not knowing where to start. Do business owners usually write their own programs, purchase a set of programs from a vendor, or hire a consultant?"
R. L., Baltimore, MD

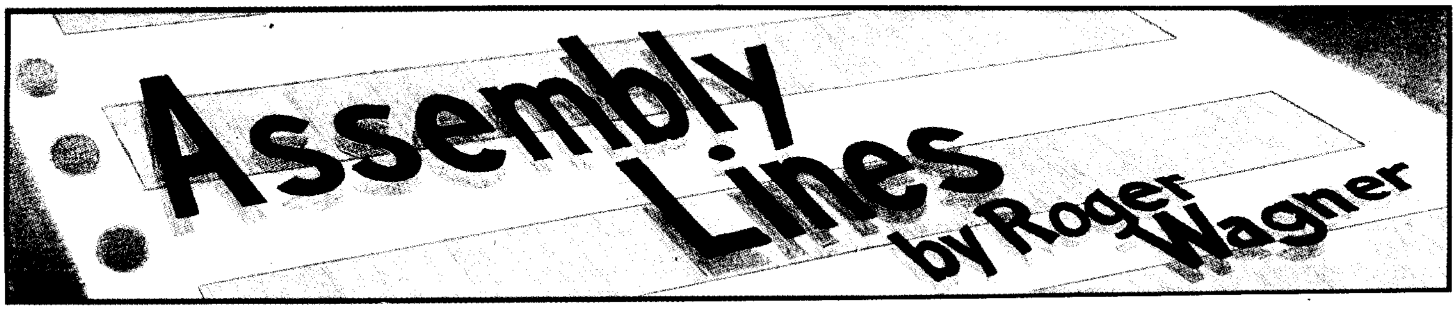
Whether or not you should be using your Apple in your business is a question only you can answer—considering, of course, the potential problems that might follow when you remove the entertainment from your home. The question is a common one, yet there's no common answer. Each of the methods you suggest has been used as the start-up step of a business. In each case, the desired application was successfully developed. The outcome depends a great deal on the people involved. Some Apple owners would rather do anything than write a computer program. Others would not even consider the alternatives.

Our recommendation is to purchase a generalized package (for example *VisiCalc*) and experiment with it. This will give you some experience with a vendor-supplied package, some experience with customizing an application, and practice using a business-oriented tool.

We assume you know how your business works and are particularly competent in the area you're considering for computerization. If you have no knowledge of accounting, it wouldn't be wise to begin with an accounting package or application. If it's your first business application on a computer, start small. Create a mailing list, record expenses, or generate a monthly income report.

Once you're familiar with what's involved, you'll be in a better position to select from the alternatives that you mentioned. My personal ranking would be (1) purchase from a vendor, (2) write it yourself, and (3) hire a consultant. Some of the material discussed in this month's tutorial should help you plan future computer applications.

For Your Bookshelf. From time to time, we'll suggest books that could be valuable additions to your business bookshelf. We'll include business books, computer books, and books of general interest. As a start, we recommend a glossary to help you deal with some common computer jargon: *The Illustrated Computer Dictionary*, by Donald Spencer, published by the Charles E. Merrill Publishing Company. A paperback, the book retails for $6.95. We think you'll find it quite helpful. ❑■



Everyone's Guide to Assembly Language, Part 10

COMMANDS COVERED SO FAR:

JMP	LDA	LDX	LDY	TAX
JSR	STA	STX	STY	TAY
RTS	INC	INX	INY	TXA
NOP	DEC	DEX	DEY	TYA
—	CMP	CPX	CPY	PHA
BEQ	BNE	BCC	BCS	PLA

Plus these addressing modes:

Immediate	Absolute
Zero Page	Implicit
Relative	Indexed

Indirect Indexed
Indexed Indirect

This month's topic is the basic math operations of addition and subtraction in machine language. To an extent, we've already done some of this. The increment and decrement commands (INC/DEC, and so on) add and subtract for us. Unfortunately, they only do so by one each time (VALUE + 1 or VALUE − 1).

If you're really ambitious, you could, with the commands you have already, add or subtract any number by using a loop of repetitive operations, but this would be a bit tedious, not to mention slow. Fortunately, a better method exists. But, first, a quick review of some binary math facts.

Earlier in the series, we discussed the idea behind binary numbers and why they're so important in computers. In case you missed it, or your memory has faded, here's a fresh look.

By now you're certainly comfortable with the idea of a byte being an individual memory location which can hold a value from $00 to $FF (0 to 255). This number comes about as a direct result of the way the computer is constructed and the way in which you count in base two.

Each byte can be thought of as being physically made up of eight individual switches that can be in either an *on* or *off* position. We can count by assigning each possible combination of ons and offs a unique number value.

In the following diagrams, if a switch is off, it will be represented by a 0 in its particular position. If it's on, a 1 will be shown. When all the switches are off, we'll call that 0.

In base two, each position of the byte is called a bit, and the positions are numbered from right to left, from 0 to 7.

The pattern for counting is similar to normal decimal or hex notation. The value is increased by adding 1 each time to the digit on the far right, *carrying* as necessary. In base 10, you'd have to carry every tenth count, in hex, every sixteenth. In base two, the carry will be done *every other time*!

So . . . the first few numbers look like this:

```
Hex   Decimal          Binary
$00      0      0  0  0  0  0  0  0  0
$01      1      0  0  0  0  0  0  0  1
$02      2      0  0  0  0  0  0  1  0
$03      3      0  0  0  0  0  0  1  1
$04      4      0  0  0  0  0  1  0  0
```

Notice that in going from 1 to 2, we would add 1 to the 1 already at the first position (bit 0). This generates the carry to increment the second position (bit 1). Here is the end of the series:

```
$FD     253     1  1  1  1  1  1  0  1
$FE     254     1  1  1  1  1  1  1  0
$FF     255     1  1  1  1  1  1  1  1
```

Now the most important part. Observe what happens when the upper limit of the counter is finally reached. At $FF (255), all positions are *full*. When the next increment is done, we should carry a 1 to the next position to the left; unfortunately, that next position doesn't exist!

This is where the carry bit of the status register is used again. Before, it was used in the compare operations (CMP, for instance), but, as it happens, it is also conditioned by the next command, ADC. This stands for add with carry. When the next step is done using an ADC command, things will look like this:

```
                                          Carry
$100     256     0  0  0  0  0  0  0  0     1
```

The byte has returned to a value of 0 and the carry bit is set to a 1.

We discussed the wrap-around to zero earlier, with the increment/decrement commands, but we didn't mention the carry. That's because the INC/DEC commands don't affect the carry flag.

However, the ADC command *does* set the carry flag. The carry will be set whenever the result of the addition is greater than $FF. With the ADC, you can make your counters increment by values other than one—rather like the FOR I = 1 to 10 STEP 5 statement in Basic. But ADC is used more often for general math operations, such as calculating new addresses or screen positions, among a wide variety of other applications.

Whenever ADC is used, the value indicated is added to the contents of the accumulator. The value can be stated either directly by use of a immediate value or with an indirect value as you've done previously with commands such as LDA.

Important Note: ADC only sets the carry if there is an overflow past $FF. It *does not* clear it (set it to zero) if the result is $FF or less. Also, you'll notice that ADC did stand for add *with* carry. This means that if the carry bit is set before the addition, an extra unit will be added along with the value you wanted. This is to accommodate two-byte numbers such as addresses, which will be discussed shortly.

Example:

Accumulator	Value	Carry Set?	Result
$80	$05	no (0)	$85
$80	$05	yes (1)	$86

What all this means is that the carry bit *must* be cleared *before* the ADC operation. The command for this is CLC, for clear carry.

Here are some sample programs for using the ADC. Note the use of the CLC before each ADC.

```
**************************
*  SAMPLE PROGRAM 1  *
**************************
*
  OBJ $300
  ORG $300
*
N1 EQU $06
N2 EQU $08
RSLT EQU $0A
*
START LDA N1
  CLC
  ADC N2
  STA RSLT
END RTS
```

```
**********************
* SAMPLE PROGRAM 2  *
**********************
*
  OBJ $300
  ORG $300
*
N1 EQU $06
RSLT EQU $0A
*
*
START LDA N1
  CLC
  ADC # $80
  STA RSLT
END RTS
```

```
**********************
* SAMPLE PROGRAM 3  *
**********************
*
 OBJ $300
  ORG $300
*
N1 EQU $06
INDX EQU $08
RSLT EQU $0A
TBL EQU $300
*
START LDA N1
```

```
      LDX INDX
      CLC
      ADC TBL,X
      STA RSLT
 END RTS
```

```
***********************
* SAMPLE PROGRAM 4  *
***********************
*
      OBJ $300
      ORG $300
*
N1 EQU $06
INDX EQU $08
RSLT EQU $0A
PTR EQU $1E
*
START LDA #$00
      STA PTR
      LDA #$03
      STA PTR+1
      LDA N1
      LDY INDX
      CLC
      ADC (PTR), Y
      STA RSLT
END RTS
```

In the first two programs the value in N1 is added to the contents of N1 and the immediate value $80. Note the CLC before the ADC to ensure an accurate result. This result is then returned in location $0A. This routine could be used as a subroutine for another machine language program, or it could called from Basic after passing the values to locations 6 and 8.

The latter two programs are a more elaborate example where the indirect modes are used to find the value from a table starting at $300. In program 3, an index value is passed to location $08. That is used as an offset via the X register. In program 4, the low-order and high-order bytes for the address $300 are first put in a pair of pointer bytes ($1E,$1F) and the offset is put in the Y register.

The main disadvantage to all these programs is that we're limited to one-byte values for both the original values and the result of the addition.

The solution is to use the carry flag to create a two-byte addition routine. Here's an example:

```
*****************************
* SAMPLE PROGRAM 5A *
*****************************
*
   OBJ $300
   ORG $300
*
N1 EQU $06
N2 EQU $08
RSLT EQU $0A
*
START CLC
   LDA N1
   ADC N2
   STA RSLT
   LDA N1+1
   ADC N2+1
   STA RSLT+1
END RTS
```

```
*300L
0300-   18          CLC
0301-   A5   06     LDA    $06
0303-   65   08            $08
0305-   85   0A     STA    $0A
0307-   A5   07     LDA    $07
0309-   65   09     ADC    $09
030B-   85   0B     STA    $0B
030D-   60          RTS
```

Notice that N1, N2, and RSLT are all two-byte numbers, with the second byte of each pair being used for the high-order byte. This allows us to use values and results from $00 to $FFFF (0 to 65535). This is sufficient for any address in the Apple II, although, by using three or more bytes, we could accommodate numbers much larger than $FFFF.

A few words of explanation about this program. First, the CLC has been moved to the beginning of the routine. Although it need only precede the ADC command, it has no effect on the LDA, so it is put at the beginning of the routine for aesthetic purposes. It also helps identify the overall unit as a math routine. Once the two low-order bytes of N1 and N2 are added and the partial result stored, the high-order bytes are added. If an overflow was generated in the first addition, the carry will be set and an extra unit added in the second addition. Note that the carry remains unaffected during the LDA N1+1 operation.

The monitor listing is given for this one so that you can enter it and then call it from this Basic program:

```
List
 0  REM MACHINE ADDITION ROUTINE
10  HOME
20  INPUT "N1, N2?"; N1, N2
30  N1 = ABS (N2)
40  POKE 6, N1 — INT (N1 / 256) * 256: POKE 7, INT (N1 / 256)
      * 256: POKE 9,  INT (N2 / 256)
50  POKE 8, N2 — INT (N2/256) * 256: POKE 9, INT (N2/256)
60  CALL 768
70  PRINT: PRINT "RESULT IS: "; PEEK (10) + 256 * PEEK (11)
80  PRINT: GOTO 20
```

The ABS() statements on line 30 eliminate values less than zero. Although there are conventions for handling negative numbers, this routine is not that sophisticated.

Many times, the number being added to a base address is known always to be $FF or less, so only one byte for N2 is needed. A two/one addition routine looks like this:

```
1  ************************
2  * SAMPLE PROGRAM 5B *
3  ************************
4  *
5    OBJ $300
6    ORG $300
7  *
8  N1 EQU $06
9  N2 EQU $08
10 RSLT EQU $0A
11 *
12 START CLC
13   LDA N1
14   ADC N2
15   STA RSLT
16   BCC END
17   LDA N1+1
18   ADC #$00
19   STA RSLT+1
20 END RTS
```

```
1  ************************
2  * SAMPLE PROGRAM 5C *
3  ************************
4  *
5    OBJ $300
6    ORG $300
7  *
8  N1 EQU $06
9  N2 EQU $08
10 RSLT EQU $0A
11 *
12 START CLC
13     LDA N1
14   ADC N2
15   STA RSLT
16   BCC END
17   LDA N1+1
18   STA RSLT+1
19   INC RSLT+1
20 END RTS
```

For speed, if a carry isn't generated on line 14, the program skips directly to the end. If, however, the carry is set, the value in N1+1 gets incremented by one, even though the ADC says an immediate $00. The $00 acts as a dummy value to allow the carry to do its job. If speed is not a concern, the BCC can even be left out with no ill effect. Program 5C shows an alternate method using the INC command. In this case, the BCC is required for proper operation.

Subtraction. Subtraction is done like addition except that a *borrow* is required. Rather than using a separate flag for this operation, the computer recognizes the carry as sort of a reverse borrow.

That is, a set carry flag will be treated by the subtract command as a *clear borrow* (no borrow taken); a clear carry as a *set borrow* (borrow unit taken).

The command for subtraction is SBC, for subtract with carry. The borrow is cleared with the command SEC, for set carry. (Remember, things are backward here). A subtraction equivalent of program 5A looks like this:

```
1  ***********************
2  * SAMPLE PROGRAM 6 *
3  ***********************
4  *
5    OBJ $300
6    ORG $300
7  *
8  N1 EQU $06
9  N2 EQU $08
10 RSLT EQU $0A
11 *
12 START SEC
13   LDA N1
14   SBC N2
15   STA RSLT
16   LDA N1+1
17   SBC N2+1
18   STA RSLT+1
19 END RTS
```

The program can be called with the same Basic program we used for the addition routine (program 5A).

These two simple ideas will prove very valuable and are well worth the time to master. With these, you have enough commands to write a great many programs. In fact, some popular commercial programs can be assembled using just the commands you've learned so far. To show the power of what's been learned, next month I'll concentrate on a specific application (disk access via RWTS) to provide some solid examples of what can be done with these commands.

FASHION

The Mad Shirter's Computer T Party!

It's a hot summer day and you are driving down California Interstate 5 counting the sheeplike power-line towers and trying not to doze off. A scruffy looking, tattooed biker zooms by wearing a T-shirt with Biff Bludgeon and the Ice Crushers printed on it. Your tires screeching, you barely miss hitting the car in front of you. On the bumper of that car is a thin rectangular sticker that asks you to "Honk If You Love Parsley."

Later in the week, you are walking through Manhattan's Central Park on your way to the Plaza Hotel. You run headlong into a brown-haired goddess wearing a T-shirt that declares: "Our Computers Put Out." You ask her if she'd like an ice cream cone, but she's heard that line before.

Walking away broken-hearted, you realize that people are trying to communicate something to you.

The Voice of the Chest. Entirely visual forms of communication abound in our society. By looking at a person's house, car, attire, and place of employment, you can glean a certain amount of information. Looking, it must be noted, is a poor substitute for the real thing: a one-on-one dialogue.

Nonetheless, one highly specialized form of visual communication, like the bumper sticker and the tattoo, is the T-shirt.

Webster's *New Collegiate Dictionary* defines a T-shirt as a "collarless short-sleeved or sleeveless cotton undershirt for men" or "a cotton or wool jersey outer shirt of similar design."

This story concerns T-shirts that have ironed, printed, or scrawled on them something to do with computers. In other words, computer T-shirts.

But first, a little history.

A Backward Glance. It is well documented in Turkey McNugget's *Cotton Comes to Town: A Brief History of T-Shirts* that, in 2750 B.C., King Sargon of the Sumerian Akkadian Empire had painted on his battle shield in the Semitic alphabet: "Mediterranean Sea or Bust." This is the first major example of a slogan being communicated visually with the use of someone's daily accouterments.

In 521 B.C., King Darius failed to vanquish the Scythians before exhausting his supplies and army. One night, without informing the victims, he took his army and left the sick and wounded in the camp to fool the Scythians, making good his escape. Broken-bodied, but not broken in spirit, those left behind wore shirts with "I Survived the Great Fink Out" embroidered on them. This is one of the most dramatic early examples of commemorating an event on apparel.

McNugget is full of examples of early T-shirt history.

In 751 A.D., Pepin Martel got the go-ahead nod from the Pope and emblazoned on his armor, "Try Pepin, the Real King." Moving from kingly advertising to simple declarative statements, we can find Cardinal Mazarin touting his philosphy during off hours in a black tanktop with "Machiavellians Rule" stitched in white on it.

McNugget also makes clear that esoteric messages have found their ways onto people's backs. A serious John Milton boasted, "I Fried the Infernal Serpent" on his.

In the 1760s, people were confused by James Watt, father of the steam engine, sporting on the back of his waistcoat the phrase, "I'd Rather Be Sailing." He was prophetic in many ways.

Facetious and deliberate confusing of messages in a clever and humorous fashion is evident in Charles Darwin's classic rib, "Never Give an Ape an Even Break."

A rare item from the early years of this century is a shirt

BY DAVID HUNTER

meant to glorify what turned out to be the Wright Brothers' biggest failure. The blue cotton shirt has a two-color illustration and the catchy line, 'You'll Believe an Elephant Can Ride a Bike."

Finally, we come to Immanuel Velikovsky. In support of his incredible theories concerning Mars and Venus being shot from the Red Spot of Jupiter into their present orbits, he had "You've Come a Long Way, Venus" printed on his T-shirt.

So much for a little history.

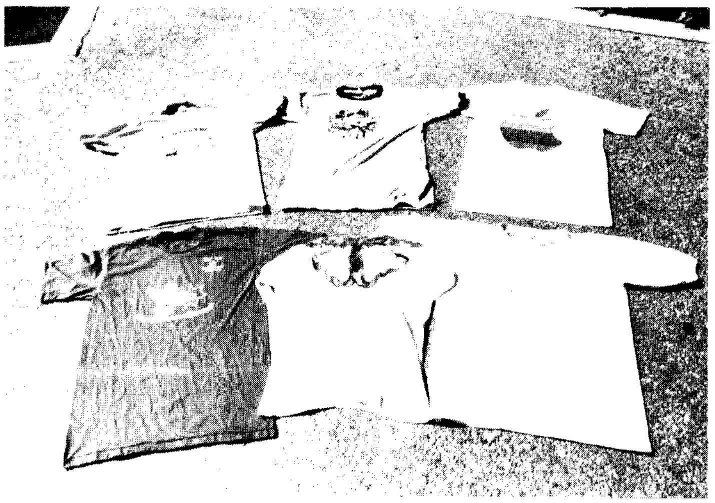
Seriously, folks. As king and cardinal of old declared their viewpoints on their chests, so do millions of people today. The variety of messages found on human fronts and backs is staggering. T-shirts glorify just about everything.

For the Aficionado. To get back on the track, the best place to look for computer T-shirts is at a computer convention. These gatherings offer a gluttonous concentration of even the most esoteric devils—T-shirts unavailable in any store or from any company. Prospective T-shirt wearers can get good ideas for custom designs by merely walking around with their eyes open.

Getting a custom design pressed or silk-screened on a single T-shirt costs anywhere from $7 to $15, depending on how fancy you want to get. You can cut down the cost by supplying your own unadorned T-shirt.

For those too lazy to make their own, here are brief descriptions of shirts available from some of your favorite software companies.

T-Shirt Shopper. Creative Computing, 39 East Hanover Avenue, Morris Plains, NJ 07950, offers eight different computer T-shirts, decorated with everything from the company name over a computer portrait of Albert Einstein to the Program Bug from the book *Katie and the Computer*. Specify S, M, L, XL, or children's sizes. Shirts are $6 plus $.75 handling. Two or more shirts can be charged to VISA, MasterCard, and American Express.



Left to right, top: Robot Rabbit and Computer Bum from Creative Computing; the original Apple shirt. Bottom: Broderbund mascot shirt; custom-painted Tempting Apples shirt; Sirius's Space Eggs iron-on ironed on.

Rainbow Computing, 9719 Reseda Boulevard, Northridge, CA 91324, offers a shirt with the curious "Our Computers Put Out" on the front and the company's colorful rainbow logo on the back. Shirts come in S, M, L, XL, and children's sizes, at $5 plus $2 handling.

Shirts in myriad colors from Broderbund, Box 3266, Eugene, OR 97403, show the company's mascot in a karate pose with "Always Wear Software" positioned underneath. Broderbund's shirts are $6.95 plus $1 handling and come in S, M, L, and XL; VISA or MasterCard okay.

"I Survived the Night at Mystery House," announce the shirts from On-Line Systems, 36757 Mudge Ranch Road, Coarsegold, CA 93614, embellished by a reproduction of the eerie old mansion you've seen in their ads. "Hi-Res Adventure by On-Line" is the message for those walking behind you. Same sizes, several colors, all for $6, nada for handling.

Sirius Software suggests you celebrate victories, but they don't actually sell a T-shirt. Instead, they give each purchaser of *Space Eggs* a decal that you can iron onto your own shirt and thereafter put the world on notice with, "I Fried the Space Eggs," surrounded by some of that game's nasty little critters.

C.A.S.H. (Computer Assisted Shirt Hucksters), 155 Yale Road, Menlo Park, CA 94025, offers a handsome shirt commemorating the sixth annual West Coast Computer Faire. Standard sizes, $8.

From Apple Computer Inc. come two styles of T-shirt for adults or children without a written word on them. Base shirts come in several colors. In the old style, each shirt is adorned simply with a large rainbow-colored bitten Apple apple. Prices range from $5.95 to $7.95 depending on size. The new style shows a smaller apple to one side of the front on an Apple rainbow of stripes that gird the chest. Women's sizes are $9; children's and men's, $7. Many Apple dealers carry these or can order them for you.

There are dozens—maybe hundreds—of computer-related T-shirts to be found if you keep your eyes open.

Users Speak. Why wear a computer T-shirt? George Snoozeoff of Slipshod Software commented, "I want people to know exactly where I'm coming from."

Beer B. Ash of Moonshine Computing offers that "it's a great way to meet women programmers in the local bars."

Figby Leaf boasts, "It's the only thing I'll wear."

Patsy Coder claims it's terrific for breaking the ice at a party or in the library, but she advises, "Don't ever wear them in local bars."

Whatever one's reasons, it cannot be denied that T-shirts go hand-in-hand with summer. The weather's warm, the trees are in bloom, school's out, and vacation time is in the air.

Summer's here and the time is right for wearing computer T-shirts.

MARKETALK

News

☐ **Inmac** (Santa Clara, CA) has announced the Inmac Plus premium brand floppy disk. The news is the guarantee: Inmac promises to replace at no charge any Inmac Plus diskette that crashes—ever. The industry's first lifetime guaranteed flexible diskettes are made possible through a new annealing process that controls dimensional changes within the substrate due to variations in temperature and humidity. In addition, the diskettes are treated with a flexible coating to combat oxide flaking. In eight-inch and five-and-one-quarter formats, $6.95 and $5.25 respectively per disk. Discounts available for bulk orders.

☐ A full general ledger, payroll, job cost accounting, and subcontractor system constitutes the **Charles Mann & Associates** (Yucca Valley, CA) package for the construction industry. *The Construction Accounting System* also includes a report preparation system for job bids, proposals, and annual reports. Program's special subcontract features locate jobs under contract, change order amounts, determine payment to date by job, and figure costs remaining. 48K, DOS 3.2 and 3.3, 130-column printer. $549.95.

☐ **Peripherals Plus** (Morris Plains, NJ) presents *Super Paddles*. Each of two paddles consists of a high precision linear potentiometer and a half-inch diameter, industrial quality pushbutton mounted in a sturdy metal case. Set includes a five-foot cable. $39.95. The same company's *Super Joy Stick* is suitable for high precision professional applications as well as game playing. $59.95.

☐ **Computer Station** (St. Louis, MO) offers *Boot Button*, which enables Apples with 3.3 DOS to be switched to 3.2 by the push of a button. Add-on board fits onto the Disk II controller card. $34.95.

☐ A rapid retrieval data base for Apple literature (through 1980) has been developed by **Connecticut Information Systems Company** (Bridgeport, CT). *Apple Data Base* scans five years of accumulated references and locates Apple-related articles. Each data base entry provides article name, author, periodical, issue date, and page number. Applesoft ROM, 48K. $60.

☐ **G. S. Computer Enterprises** (Ann Arbor, MI) offers a 32K expansion board featuring two 16K banks of buffered RAM on a single plug-in board. Compatible with Integer Basic, Applesoft, Pascal, Fortran, and SoftCard. DOS 3.2 and 3.3 16K version can be expanded to 32K. 32K: $189; 16K: $159.

☐ Second edition of *CP/M Software Index*, from **Small Systems Group**, (Santa Monica, CA), lists 740 CP/M programs for the Apple offered by 248 vendors. Lists more than twice as many programs and vendors as last year's edition. $6.

☐ **On-Line Systems** (Coursegold, CA) announces *The Expediter II*, an Applesoft compiler. By converting Applesoft programs to machine-language object code, *Expediter II* increases execution speed by a factor of three or more. 48K, disk. $99.95.

☐ For *VisiCalc* users, independent newsletter emphasizes developments in business and professional applications. Write to *VisiNews*, Box 341, Kings Park, NY 11754 for a sample issue; subscriptions are $15 per year.

☐ A home finance system, *Money Maestro*, is the first microcomputer product of **Innosys Incorporated** (Berkeley, CA). Program offers the sole proprietor, property owner, or two-income family finance planner a sophisticated single-entry accounting procedure. Written in Forth. 48K, CP/M. $200.

☐ **Gibson Laboratories** (Irvine, CA) has developed a hi-res light pen compatible with all available languages for the Apple II. Use for menu selection, business graphics, drafting and architecture, circuit analysis, word processing, and game playing. The *LPS II* promises speed and versatility for animators and artists because of its unique high-speed, video-synchronized search technique. Installs on Apple motherboard. $285.

☐ **SSM Microcomputer Products** (San Jose, CA) offers the *ASIO*, a plug-in board for connecting serial-driven peripherals to the Apple. Board has connectors, one for a DTE devices, such as a printer, and one for a DCE devices, such as a modem. It also provides an RS-232 serial interface and basic jumper-selectable communications baud rates from 110 to 9,600. Features include automatic line feed on carriage return; automatic carriage return whenever a selected line length is exceeded; and variable delay after transmission of carriage return (for slow printers). Assembled and tested. $139; kit: $109.

☐ **Automated Simulations** (Mountain View, CA) presents *Dragon's Eye*, an EPYX game that challenges the player to a search for the magic jewel held by an evil magician who derives his power from the gem. Object of the game is to recover the jewel, which is hidden somewhere within one of seven provinces, and return it to Fel City, where the journey begins. The player assumes the identity of one of sixteen magical characters who are gifted with abilities such as flying, healing, and traveling in time. 48K, Applesoft, disk. $24.95. The worm has turned in another release from Automated Simulations. In *Crush, Crumble and Chomp,* you play the role of your favorite monster in any of more than a hundred scenarios. You can choose between six man-eating monsters, but you'll always have the same basic need: to satisfy an enormous appetite by eating your opponents. In the meantime, you must battle National Guard tanks, infantry, helicopters, and a team of mad scientists. You also choose your game objective from among five: to destroy as many buildings as possible; to destroy combat units but spare citizens; to eat your way to a new high; to destroy everything in your path; or merely to survive the longest time. If you tire of these, you can create your own monsters. If this all sounds totally weird to you, you are not alone. But if you happen to have a whole bunch of hostilities stored up this month, this game may be tailormade to help you out. 48K, disk. $29.95.

☐ **Aurora Systems** (Madison, WI) presents *VersaCalc*, which expands *VisiCalc*'s display screens. *VersaCalc* also shows how to do conditional testing, display error messages, interchange columns and rows, and automatically execute a string of up to 255 *VisiCalc* commands with only four keystrokes. 32K. $100. *The Executive Secretary*, Aurora's word processor, features professional speed typing, page numbering, editing, automatic addressing for envelopes, and shift key conversion for lower case. Works with forty-column or eighty-column screen. 48K, ROM Applesoft. $250. A third Aurora offering, *Hebrew II*, puts Hebrew characters on the screen from right to left, allows full cursor movement and editing, and, with a printer, it can produce labels, posters, and correspondence in Hebrew. 48K. ROM Applesoft. $60.

☐ **High Technology Software** (Oklahoma City, OK) announces *Job Control System*, a system for job management and administration. Program consolidates and files labor hours, material costs, outside service costs, production quantities, and shipped quantities, helping managers determine productivity and up-to-the-minute job status data. Requires 132-column printer capable of form feed. 48K, Pascal. $750.

☐ Another aid to management intended for tracking costs and labor hours comes from **Westware, Inc.** (Ontario, OR). *Job Costing Module* can be used on a standalone basis or integrated to work with Westware's full accounting system, which includes general ledger, accounts payable and receivable, inventory, and payroll. 48K, Applesoft, DOS 3.3. $345.

☐ **Phase Zero Software** (Tucson, AZ) introduces *ASCOMP 2.5*, an Applesoft Basic compiler, which works in true machine code, but uses some functions of Applesoft ROM for a run-time package of about 1K byte. All Applesoft code (except Resume) is compiled. Intermediate use of the disk maximizes compiled program size. Options include creation of machine language subroutines that can be called from a Basic program (multiple entry points), split compiled code above and below the hi-res screen(s), and control-C stop of compiled programs. $85.

☐ **Software Technology for Computers** (Belmont, MA) has announced a new color graphics package that lets users create, erase, change, store, and retrieve graphics using either or both drawings and text. The hi-res *Coloring Board* offers six colors and can generate arcs, angles, circles, squares, and other geometric shapes. TK. $60.

☐ **Omega Peripherals** (Columbia, MD) offers *Omega Joy Stick*, now generally available after a year of exclusive availability through custom order. The handheld, all-metal joy stick is cased in hard plastic. Stick is made for smooth action and has a large handle ball for sure grip. Generous metal push buttons are designed for comfort during intense game playing. Nonself-centering feature allows users to switch between joy stick and buttons without losing desired screen position. Trim potentiometers for extra fine tuning optional. $59.95.

☐ The jobs of serial interface, parallel output interface, and real-time clock/calendar have been combined on the *CPS Multifunction Card* by **Mountain Computer** (Scotts Valley, CA). You configure CPS from disk, setting up function parameters that may be changed with keyboard control commands. Serial and parallel output can be used simultaneously from CPS. Separate CPS functions can be assigned to Apple's various slots, into which card does not have to be plugged. Clock/calendar has time capability of from one second to ninety-nine years and is compatible with MCI Apple Clock time-access programs. Introductory price, $239.

☐ *Dual DOS ROMs* from **Soft CTRL Systems** (West Milford, NJ) saves effort of booting from one DOS to another, Utility plugs into Romplus or Andromeda ROMBoard, embedding both ROMS in memory. CALL command toggles between either DOS according to your requirements. DOS toggling doesn't affect any programs in memory. Message at lower right of screen tells you which DOS is being used. Utility can't be used for booting 3.2 disks. 48K, DOS 3.3. $49.95.

☐ According to **Apple**, you'd have to have a mini to get the mailing list capabilities you can now find on Apple's new *Mail List Manager*. Now, all you need is an Apple III. *List* stores, sorts, edits, and prints mailing labels and phone lists in entirety or selectively by zip, name, or any other data. Disk max is 960 labels, which can be sorted in less than two minutes. Labels can be defined up to six lines per entry and may be sorted on any two of these fields. Further information, however, can be stored with the disk entry for each item and omitted in printing. Disks can be merged to create a single larger mailing list. For printing labels, user can specify label size, number of labels across page, and space between labels. Backup disk included. Apple III, 128K, extra drive. $150.

☐ *AppleGraph*, from **Business & Professional Software** (Cambridge, MA), which was purchased recently by **Apple**, is now available under the Apple logo with a slightly changed name and probably some enhancement as well. The Apple product is called *AppleGraphics II*. The package enables you to design two-and three-dimensional graphics, then to see the objects in your designs from any angle. It also allows full-screen viewing of any portion of a drawing. *AppleGraphics II* programs can be written in Pascal or Fortran; printing can be achieved through most advanced plotters. 48K, Language System, disk. $95.

☐ **Qume Corporation** (San Jose, CA) has readied three models of their *Sprint 9* daisywheel printing terminal. Direct-drive mechanism meets high standard of print accuracy. DuPont-designed belt and pulley drive mechanism lessens printer chassis tension and increases life of parts. *935* model is a limited function, RCV-only terminal that prints thirty-five characters per second. $1,995. With interactive KSR time-sharing capability, *935* model is $2,095. Faster full control models, the *945* and *955*, can interface with minis as well as Apples. $2,455 and $2,555.

☐ From **Advanced Management Strategies** (Atlanta, GA), *Target* calculates and analyzes past and future business activities with ability to print or display all data entries, report specifications, and perform calculations for a project. Package is sold through **Westico** (Norwalk, CT). SoftCard, two disk drives, 48K, DOS 3.3. $195.

☐ A workshop for eight to eighteen year olds is being conducted throughout the summer in North Hollywood, CA. Workshop features instruction on Apples and other micros in a relaxed, self-paced learning environment. Main instruction will be in Basic, but the newly developed language, Logo, introduced by MIT professor Seymour Papert in his book *Mindstorms*, will also be taught. Students meet three hours daily for two weeks. Sessions are scheduled from July 13 to 24; July 27 to August 7; and August 10 to 21. Student-teacher ratio is twelve to one;

GOTO 42

SOFTALK INTERVIEW: WADE and NANCY HARRIS

a candid conversation with a couple about their much maligned occupation and the impact of computer technology on their business and family.

News travels fast in this competitive industry. Author **Jim Salmons** *had just left his position in marketing management with a major software publisher in Baltimore to start his own business. We understood his desire to explore innovative microcomputer marketing strategies; but wouldn't he have to pay the bills until his own business became profitable? So, when the* Softalk *data base of interesting people doing unusual things with their Apples turned up Wade and Nancy Harris, a couple in the chimney sweeping business in Ellicott City, Maryland, we called Salmons. After feigning, "Let me see if I can work this into my busy schedule," he jumped at the chance to make some spare change. Here is Salmons's report:*

"Somewhere between chimes and chimpanzees I found them: chimney sweeps. An even dozen sources were cited in the subject index at the Eisenhower Library of Johns Hopkins University. A good investigative reporter does homework before raking muck or, in my case, soot.

"Sure, Softalk *wants a cutesy Us-and-Our-Apple profile of this couple using their computer in an unusual business. But what about those saltpeter-scabbed knees and elbows of the consumptive innocent climbing boys whose flue-scraping labors are repaid by abuse and neglect at the hands of their sadistic masters. Was Dickens's* Oliver Twist *to be the last expose on these occupational abuses? Or would we have to wait till the evening news brings us live coverage of the indelicate recovery of the suffocated body of a child forced up a too-narrow flue?*

"My resurrected bleeding heart surged with the adrenaline of moral outrage. The sudden flow of blood to my head refreshed memories of my early investigative interview training in journalism school—not my professors' lectures on pyramid writing style, but the subliminal training from obligatory scans of Playboy *interviews before copiously studying details of the photographic art.*

"These interviews cut below the surface, providing psychosocial insights into the souls of the interviewees. The good apples show as good; the bad, rotten . . . all in their own words. This would be the gauntlet to which I would subject Wade and Nancy Harris. And this is how I found myself in the bowels of Hopkins Library on a glorious Baltimore Memorial Day weekend.

"A quick scan of the scant literature confirmed my worst fears. Only one contemporary author had seen fit to follow up on this significant social disgrace of the early nineteenth century. The abuses of the English and European sweeps had been well chronicled a century earlier but none since until, in 1957, George Lewis Phillips published his scathing historical expose, American Chimney Sweeps *(The Past Times Press, Trenton, NJ). For more than a century we had hidden this ugly thread under our colonial round-braid carpets until Phillips had the courage to weave it back into the fabric of our American collective consciousness. Unfortunately, his book didn't make it to paperback; consequently, we didn't see these tidbits on television's 'Bicentennial Minutes.'*

"Fire was a major environmental hazard in most colonial towns. The log and thatch houses huddled together to afford protection from Indian attack were volatile prey to any blaze started by a careless neighbor's chimney fire. The cure, huddled houses, was worse than the disease, Indian attack.

"So combustible a hazard were these unplanned communities that the local governing councils took it upon themselves to create subsidized positions called **town master sweeps**. *It was as if the councils thought giving an official title to the job would erase the social stigma chimney sweeping carried in the minds of the early settlers who had witnessed scandal and degradation within the trade before coming to the New World. In a land where there were so few people for too many jobs, would you expect anyone to volunteer for a job for which your neighbors hated you and in which a major occupational health hazard was cancer of the scrotum?*

"So it was in 1662 when the town fathers of Salem, Massachusetts, appointed John Milke the first American town master sweep. He and others in similar positions in other colonial towns were empowered to obtain fees for services rendered as well as to fine neighbors who harbored dirty flues. This enviable catch-22 of inspector/contractor was not enough to overcome the degrading work and social rejection, so many master sweep positions went unfilled.

"For nearly a century, the houses burned while most towns constantly advertised for sweeps in the help-wanted sections of the early colonial press. Some towns even foreshadowed the draft and jury duty by passing legislation appointing sweeps

"If one of the little beggars balks, I just give him a shot to the buns with this. That gets them up the chimney."

"This is the soot the boys coughed up onto the keyboard when we invited them in from the shed to play Space Eggs.*"*

"When the boys are too sick to work, I tie a rope around the neck of a goose and drop it down the chimney, like this."

Softalk photos

"The boys seem to last longer if we hose them down once a month or so, unless it's below freezing in the shed. . . ."

by lottery. Only the wealthiest citizens could afford the prohibitive fine for refusing the appointment.

"Other towns encouraged private sector sweep monopolies in which the sweep ventures split fees and fines with the town treasury. The monopolies were allowed to establish their own collection agencies; the collectors were as large as the climbing boys were small.

"In some lucky towns, there were some unlucky souls who couldn't make it at any other trade, so they took up chimney sweeping. While working conditions were poor, the pay was worse. Many made ends meet by moonlighting as carpet beaters, white washers, wood choppers, even cesspool emptiers.

"In rural towns too small to attract their own sweeps, a rather foul method of chimney cleaning developed. One simply tied a rope around the neck of a large goose, dropped the bird down the chimney, pulled it back up, and repeated the operation several times. The distressed bird's flapping wings loosened the soot. The blacker the bird, the cleaner the flue. This method gave new meaning in colonial times to the age-old expression, 'Wanna get goosed?'

"Other sweeping methods were less flamboyant. In the earliest times, colonial houses were one story high with large diameter chimneys. For these, a sweep had only to stand on a ladder and use a long handled broom. At these homes, master sweeps were notorious for saying such things as, 'Hey, I'll bet TB really takes it out of you. Why don't you take a breather. I'll get this one and you can do the next.'

"Of course, the next was invariably a newer, multistoried residence where unschooled builders constructed the most convoluted flues imaginable. For these, the master would suggest that the climbing boy, with scraper and broom in hand, crawl up the flue to clean it. These suggestions took the form of swift kicks or pin pricks to the boy's bottom. If a boy became stuck in a flue, masters found that a small straw fire in the hearth had a persuasive effect in coaxing the boy up and out.

"If a chimney was particularly narrow, the boys got a rest. A canvas bag of bricks or a bundle of sticks was forced through the flue. This urban variation on the goose method was more humane but less effective. In this and all cleaning methods, the master pinned a blanket across the hearth so as not to soil the homeowner's rooms and furnishings.

"At the end of a job, the climbing boys gathered all the debris in the blanket to take back to their shed. There they sifted cinders from the soot to sell as fertilizer. They were then allowed to sleep, unwashed, in the shed, using the hearth blanket to protect against the cold

"If this sounds scandalous, it actually became worse. America's most heinous management solution to the need for rapid national growth was the unconscionable acceptance and exploitation of slavery as a labor source. All manner of work considered too disgusting for 'humans' was relegated to black slaves.

"And so it was that, for almost a century, white America's chimneys were cleaned by black climbing boys. With the investment they had in the developing society, it was little wonder that the blacks were notoriously ineffective sweeps. Unfortunately, it also explains the blind spot America had to the growing social consciousness that rose against the abuses of the trade in England and Europe.

"In the late eighteenth century, Jonas Hanway was instrumental in forming a society to lobby against labor abuses and lobby for support of technological developments to replace climbing boys. There was no comparable organized group in America, though isolated humanitarians did show some concern.

"An example of 'reform' in American trade comes from my hometown of Baltimore. In the 1790s, a white master sweep, John Conrad Zollikoffer, advertised in the Baltimore Daily Repository *that parents consider indenturing their sons as sweeps for a period of five to seven years. In return, at the end of their servitude, Zollikoffer would finance the boys' training to enter more elevated professions. Come on, John, after kicking these kids' butts up and down the flues of Baltimore's rowhouses for seven years, did you really expect any of them to have the vigor to enter vocational school?*

"So it was until mid nineteenth century, when the first era

of American chimney sweeping technology was ushered in. By 1830, at least six patent applications for sweep devices were pending. Some were original devices, others were variations on the Smart and Glass devices invented in England in the late 1700s. Master sweeps who incorporated these devices into their work were called fluonomists, *technological sweepers. The transition between old and new was rocky, in-fighting and professional jealousy abounded.*

"And here is where the literature breaks down. Phillips's book was a recent addition, but it was purely an historical retrospective. What's needed is an investigative report. What's happening out there in America's chimneys? Have the technological innovations been incorporated into the sweep trade? Or is America still harboring wasted, abused children in the hidden sheds of alleged good citizen master sweeps? Are Apple-using Wade and Nancy Harris cyberfluonomists or debased climbing boy molesters?

"With these questions burning, I arrived in Ellicott City, a quaint town frozen in time, nestled between the throbbing metropolis of Baltimore and the pristine crispness of Columbia, urban developer James Rouse's vision of the city of the future. As I walked the streets, the sense of beauty and history almost distracted me from my purpose: I was about to rip the lid off this lazy burg and expose the maniacs in its midst. . . ."

WADE: Hey, what the . . . You down by the shed, may I help you?

SOFTALK: Oh . . . a . . . Hi! Jim Salmons, *Softalk* magazine. You must be Wade Harris.

WADE: Right.

SOFTALK: We talked earlier about doing an interview, remember? I know this looks funny, me breaking in your shed and all. But I dropped a penny when I got out of the car and it rolled down here, so. . . .

WADE: It rolled through the grass?

SOFTALK: . . . Yeah, ah, well it was only a penny. . . . Say, how about we go on in the house; I'd like to talk and meet the rest of the Harris clan. [*I'll get photos of the wasted little climbing beggars later.*]

WADE: Sure. Can I get you anything? Iced tea, soda, beer? You must be tired after your ride. . . .

SOFTALK: [*So it was. I got inside and what I thought was going to be a huge scandal turned out to be a fascinating discussion with what, in many ways, was a typical suburban American family. What wasn't so typical was the sideline they chose for making extra money and the fact that they are experiencing the excitement of becoming microcomputer literate together.*]

SOFTALK: Thanks, Wade, for holding up those props and reading the little quotes for the photo teasers. I'm sure they will attract a lot of readers. Now, could you introduce your family?

WADE: My son, Mark, is fourteen and he's been helping me ever since I started this part-time business three years ago. He helps me carry equipment, hands me brushes, helps me in and out of the air respirator and long gloves, sets up the vacuum, and now he even cleans some of the easier chimneys and fireplaces.

Nancy holds down the operation back home, answering the phone, setting schedules, keeping the business records. In a service business like this, your customer contact is critical. She does a great job in a professional and pleasant manner.

SOFTALK: I can attest to that. Nancy, you were most helpful in arranging this interview.

NANCY: We're really excited about being in the magazine.

SOFTALK: You mentioned that this is a part-time business, but it sounds like full-family involvement.

WADE: Yes. It's part-time work for all three of us. And during our main season from about Labor Day till Christmas, I guarantee we collectively put in a hundred hours a week or more. Nancy has even gone part-time with her office job in an insurance company.

NANCY: Right. Sometimes it gets to be a hectic home around here. Come Labor Day, the phone starts ringing. We have to turn work away. Some people get frustrated when we put them on a seven-week waiting list, but there's a limit to how many chimneys we can clean. So I only schedule so many a week.

WADE: It's about fifteen tops on weeks when I work my regular job. In the fall, when I take most of my vacation time to sweep, we do many more. It's still less than the demand. I could probably hire and train some helpers and make it a full-time family-run business. But that's not our goal.

I work for the government in the procurement of mainframe computer systems for health care agencies. It's satisfying head-type work . . . and I've built up about twenty years of government service and benefits, which are tough to give up.

Sweeping is different. I really enjoy the physical aspects. There's something very satisfying about the activity and the sense of accomplishment when I see the difference an hour and a half of my work can make. The funny thing is, while it looks like one of the dirtiest jobs around, it's more appearance than reality. Those climbing boys you mentioned could have been clean as a whistle every night, it was just that the sanitary conditions provided them were so poor. I shower and it just melts away down the drain. My clothes clean easily too.

NANCY: So it's no big messy problem around the house. I find the contact with the customers and the administrative work challenging, yet I have time for some of my own interests . . . especially cooking, my biggest hobby.

The name of our business is Top Hat Chimney Sweeping, so many of our first-time customers call and want to know if we wear the hats and tails or if it's just a name. When I tell them it's for real, they schedule right away, especially if they have children.

MARK: People seem to have a real curiosity about us. The hats probably increase the attention we attract.

WADE: I believe if you're going to be a chimney sweep you should wear the hat. It's tradition and a trade superstition to bring good luck. So when I started the business I decided I was going to wear the hat and, in cooler weather, tails, regardless of what people might think. And you know, it's the cheapest and most effective form of advertising I've found.

SOFTALK: How much of a factor have the energy crisis and increased awareness of alternative energy sources been to your business?

WADE: It's not only significant to the amount of business, it's the reason I got into it. When the cost of heating this house went up so much, I bought a wood stove inset for the family room fireplace almost five years ago. Wood stoves are much more efficient in heating the home, but they make the chimney much dirtier. Heat stays in the house rather than going up the flue where it would burn off the condensed chemicals resulting from wood combustion. So it's much more important to clean your chimney regularly if you have a wood-burning stove.

Research shows the heat of a chimney fire can exceed three thousand degrees, which can easily set the rest of your house on fire. And sure enough, after the first year of using the stove, my chimney was extremely dirty. The closest sweep I found was in Baltimore City, and he had a backlog of several months.

"Apple . . . offered far more than any of its competition."

So, I figured there would be more and more folks in my situation and that sweeping might be an up and coming business.

While looking into the subject, I learned of Black Magic Chimney Sweeping School in Stowe, Vermont. They have an intensive training program and they sell the equipment you need. They also help you set up a business. People from all over the country go there to learn the trade. There was even a sweep working out of Alabama in my class of eighteen. You get to know each other after a week of walking on steep New England roofs. I still keep in touch with some of them. We even have our own trade journal, *The Chimney Sweep Times.* And there's a National Chimney Sweep Guild that I belong to. We're a kind of subculture.

NANCY: When Wade went down to place a newspaper ad, the editors were very interested since they had never run an ad for a local chimney sweep.

WADE: So instead of running the ad, they did a feature story about us.

MARK: And the rest is history. The phone started ringing and hasn't stopped yet.

NANCY: That really got us started. We put in the business phone, the answering machine, had brochures printed, and all.

WADE: We just didn't realize it was going to grow as quickly as it did.

SOFTALK: You've mentioned growth, but you haven't told us how your Apple figures into the business. Can I venture a wild guess that, like most growing businesses, your need for organization and efficiency created a need for the computer?

WADE: Right. I've mentioned the heavy seasonal aspect of this work. In the fall, we often have more than a hundred customers on the waiting list. It was very difficult coordinating all that paperwork. Sure enough, we started messing up—losing track of customers. I was running all over the county like a chicken with its head cut off.

So I realized we needed help. Either we needed a time-share computer service four months out of the year or—well, I'd heard of these new small computers like Apple and the TRS-80. So I started shopping around. I quickly focused on the microcomputer alternative because it looked like a wondrous instrument for keeping track of scheduling and expenses. I picked the Apple because I felt it offered far more than any of its competition.

While I knew a lot about what big computers could do in terms of massive data processing systems, I wasn't too familiar with what the small ones could do. So the salesman showed me the simple data base system, *File Cabinet,* that you get free when you buy your computer. It comes in the contributed software. In fact, that's what I currently use almost exclusively for keeping records.

NANCY: It's beautiful for what we do. After Wade shopped around, he took me to the store and showed me what we would be getting. The computer was a lot different from what I expected, but I could see that it would be a big help, especially since I was coping with most of the headaches resulting from fast growth.

WADE: First, we entered all our pending customers. I came up with a record format including standard things like name and address. Then, we started tracking when folks called, when we made appointments, and any helpful notes about the customers' chimneys.

But the absolutely most helpful thing I did was to go through my entire customer base and assign neighborhood codes. Cleaning the chimney takes a fixed amount of time, but travel is variable between jobs. So I knew I could do more chimneys if I could schedule jobs closer together.

Now, if I have a commitment to clean a chimney on a Saturday in East Columbia, I just do a search of my computer records for EC code. I take care of the longest pending ones first, then fill my day out by arranging jobs near these commitments.

NANCY: It sounds simple, but it works. Wade's averaging about one more chimney a day since we started with the computer. Now our travel expenses are down, and I don't have to deal with frustrated customers because we don't lose them anymore. And happy customers are critical to this business since so much work comes by word of mouth.

WADE: That was our first application. Of course, we've expanded our uses. We now keep all our records for taxes and accounting purposes on the computer. Our family payroll is on the Apple. That's what we've done so far with our II Plus, disk drive, and television.

SOFTALK: And what does the future hold for the business applications of your system?

WADE: You noted that we don't have a printer. That's because I had a tip on the new Epson sheet and form feed printer that is just now coming out. That's the one I want. I couldn't see spending more for one of the superduper printers than I had spent on the whole system, so I've been waiting. The Epson will be just right for what we want to do.

As soon as we get that, we'll do several things. We'll generate our invoices through the system. In fact, I've started learning to program in Applesoft, and I'm writing an invoice program now.

Then I intend to go strictly to direct mail advertising. In the three years we've been working, we now have a past customer base of close to a thousand.

NANCY: Some of those have become regular customers, but a lot more of them could be if we followed up more effectively.

WADE: We want to smooth out our season work flow. We get all the business we can handle in the fall, but at other times of the year it's slow. By using direct mail, with special pricing and refer-a-friend coupons, I believe we can build up our regular customer base. This would save the fall for concentrating on servicing new prospects who can then be encouraged to shift to regular customer status for cleaning during the slower times of the year.

"We're liable to have his-and-her computers before too long."

I can pretty accurately determine when a regular customer will need a sweep. I'll be using a form letter capability to remind them it's time to call me. I'll be cranking those letters out as soon as I get the printer.

NANCY: Once the printer goes in, it's going to be even harder for me to get to the Apple. We're liable to have his-and-her computers before too long!

I told you I really like to cook. Well, I'm working on setting up a disk file of all my recipes. I want to be able to cross-reference them so I can find things quickly and easily. So many of my friends ask for copies of my dishes—soon I'll be able to zip a recipe off the Apple when someone asks for it. It'll be great—a tremendous timesaver for me . . . besides wowing my friends. The computer is still a stranger to almost all of them.

MARK: —except the kids in the neighborhood who love to play games. You would think my friends figure our house is an arcade. I even had to tell one friend the computer broke to get a rest from him coming over to play *Invaders.*

SOFTALK: Yes, I remember that scenario when our family was one of the first to get a color television.

NANCY: Right. Between the neighbors' kids and Mark starting to learn to program at school and using the Apple for his homework, we have to schedule time for each of us to get a chance to use it. This business computer has really become a business and home computer. It's something I never dreamed of. But the longer it's here, the more uses we find for it. So we're not kidding when we joke about needing more than one.

MARK: I'd like one totally to myself. In school, we're getting more time on computers and learning more about them all the time.

WADE: —I think he still puts golf and girls ahead of computers. It's amazing how the local school system is bringing computers into the curriculum. These kids will have grown up on them. It's very exciting. Here I've worked with the old mainframes for so long, and this stuff is new even to me. I never dreamed we would have so much computing power at home in my lifetime.

SOFTALK: I can see you're all hooked. It's fascinating what you're doing as a family. An exotic family business and pioneers of personal computing, right here in Ellicott City. Thanks for being so cooperative. Any closing comments?

WADE: Well, if folks would like to know more about their chimneys and fireplaces, the bible is *The Wood Burner's Encyclopedia* by Jay Shelton, a Dell paperback. If they want to know more about computers, they should head down to their local Apple dealer.

NANCY: We're glad we did.

MARK: For sure.

MARKETALK

News

from page 32

enrollment limited. Tuition is $175; inquire about partial scholarships and reduced rates for sibling enrollment. Write to Tricia Jordan, director, 5652 Elmer Avenue, North Hollywood, CA 91601, or call (213) 769-0978.

☐ **Rochester Institute of Technology** (Rochester, NY) and the **National Technical Institute for the Deaf** offer two intensive workshops for deaf adults: Introduction to Processing and Advanced Data Processing. Both feature hands-on practice with Apples. Each course is five eight-hour days long, August 3 through 7 for the introductory course, August 10 through 14 for the advanced session. Tuition of $215 per course includes room and board. Write NTID Data Processing Department, Rochester Institute of Technology, One Lomb Memorial Drive, Rochester, NY 14623, or call (716) 475-6373—voice or TTY—for information or to register.

☐ **Strategic Simulations** (Mountain View, CA) announces games for armchair athletes, politicos, and war heroes. *Computer Baseball* lets users play with any real team or create teams of their own. Computer uses stats for each player to operate game. $39.95. In *The President Elect,* political climates of national elections from 1960 to the present are simulated. Historical or fictional candidates are judged in weekly polls and can be pitted in debates. $39.95. *The Shattered Alliance* with RapidFire, a real-time game system, lets players maneuver their units simultaneously. Magic spells can also be part of warfare. Battle ends when one army loses morale or runs from the battlefield. $59.95. Each game has solitaire capacity. 48K, ROM Applesoft, disk.

☐ **Broderbund Software** (Eugene, OR) presents (at last) *Tawala's Last Redoubt,* fourth episode in Doug Carlston's galactic sagas series in which Julian du Buque of Sparta and Tawala Mundo, emperor of the planet Galactica, compete to conquer the twenty inhabited worlds of the central galactic system. New episode introduces Benthi, female leader of the planet Farside, who rebels against the exiled Tawala. Challenge is to break secret codes and master military tactics. 48K, Applesoft. $24.95. Broderbund's also offering an outer space shoot 'em up: In *Space Warrior,* player is in a space ship protected by five orbiting shields, which enemies can penetrate only by ramming them. Object is to keep them at bay with missile fire. 32K, $24.95 The company's most recent hi-res action game, *Apple Panic,* is something altogether different. Player tries to tunnel through a five-layer labyrinth while being chased by giant menacing apples. Player must work against the clock to trap and bury a succession of clever enemy apples. 48K, $29.95.

☐ **R. H. Electronics** (Buellton, CA) offers a compact fan to keep summer Apple activities cool. *Super Fan II* clips over circulation vents on the left side of the Apple. Fan serves as electrical intermediary plugging into wall socket and Apple's power supply, making start-up more convenient. Switching on fan automatically turns on the Apple. Fan increases reliability by saving downtime, extends life of chips by keeping them cool, and reduces heat generated by extra plug-in cards. 120 or 240 volts. $69.

☐ **Information Unlimited Software** (Berkeley, CA) is publisher and distributor of *Datadex,* an interactive data base management system written by

Sonoma Software (Occidental, CA). IUS says it's the first DBMS that lets users reformat data for various purposes without rekeyboarding. Program has Soundex routine that finds data—even misspelled names—via phonetic search. Report generator makes it possible to enter mathematical formulas and calculations into report format. Hardcopy, such as columnar ledgers, reports, and mailing labels, requires at least eighty-column printer; system works with up to nine disk drives. 48K, Applesoft. $395. Introductory special during July, $295.

☐ **Program Design Inc.** (Greenwich, CT) announces complete revision of *Step By Step,* an excellent beginner's course in Basic for Apples. Included in the thirty-three-lesson course are two disks, audio cassette, and workbook. Cassette supplies spoken instruction as computer illustrates program concepts using sound, graphics, and animation. Workbook contains lesson summaries, practice problems, and sample Basic programs. $79.95.

☐ **Imagineering** (Eau Claire, WI) presents *Type-Righter* word processor for Apple III, which uses Apple III's eighty-column display to show user exactly how document will be printed. Features include ability to link files in any order and automatic envelope addressing. $195.

☐ **Geo-Compu-Graph** (Spokane, WA) adds a geophysical subpackage to its *Go-Anywhere* system. The *Gremlin* package comprises a group of programs that executes the generalized reciprocal method of interpreting seismic refraction data. *Gremlin,* like other geoscientific *Go-Anywhere* programs, is built around program modules that perform a data input or edit function or generate one of four plots required in using the GRM. In most cases, programs can be transmitted via telephone link to manufacturer's computer, minimizing software transportation and installation time and cost. Requires digital data plotter. 48K, Applesoft. $2,200.

☐ **Rainbow Computing** (Northridge, CA) announces *Super Stellar Trek,* a hi-res, real-time action game featuring one-stroke display change, improved visuals, sound effects, and ion storms. 48K, ROM Applesoft. $39.95.

☐ **Amdek** (Arlington Heights, IL) announces *Color-I,* a thirteen-inch color monitor with standard composite video signal. Features include sharp color display for graphics and sixty-four-by-twenty character display. $449.

☐ **Human Systems Dynamics** (Northridge, CA) announces *HSD ANOVA,* a variance analysis program for scientific or business applications. Program analyzes balanced designs, considering up to eight independent variables and can handle designs composed of between-subjects and within-subjects factors. Program calculates p values for F ratios, handles treatment means and standard deviations, and features versatile edit and report modules. 48K. Applesoft. $74.95.

☐ The folks at **Sirius Software** (Sacramento, CA) celebrate the first anniversary of that prolific organization with the release of a trio of action games. In *Gorgon,* player is a fighter pilot caught in a time warp. Challenge is to conquer invaders whose goal is to steal people from the surface of Earth. Points are earned for both killing the confiscatory creatures and rescuing the captured Earthlings. Nasir's latest offering is played with keyboard. 48K, $39.95. *Sneakers,* similar to the arcade game *Astro Blasters,* challenges player to successfully maneuver around a succession of creatures including mean little men in tennis shoes who try to stomp the player to death and an assortment of other beasties bent on eradicating their enemy. Many heart palpitations later, player gets promoted to a second level of challenges. Played with either keyboard or paddles. Author is Michigan college student Mark Turmell. 48K, $29.95. The company's third offering is an untitled program—which may be named *Time Warp*—that marks the latest entry in the *Star Raiders* derby. This fast-paced, head-on action game benefits from the use of a joystick and features three-dimensional graphics. Games available only on disk. 48K. Price has not yet been determined. ❏■

MARKETALK Reviews

Apple Writer Extended Features. By Paul Malachowski and Kevin Cooper. When a useful software package is unprotected, creative minds will find ways to make that package even more useful. Sometimes they'll share their methods with the rest of us.

That's what Brillig Systems has done in the *Apple Writer Extended Features* package. *Apple Writer* lacks some of the power of more recent word processors, but it's unlocked. Consequently, programs such as *Graphtrix* by Data Transforms could be developed to increase the power and value of *Apple Writer* by enabling you to format your *Apple Writer* files with chapter titles and running heads and to include graphics in your reports, calling graphic figures directly from *Apple Writer* to be printed at the first appropriate place in the text after their first mention.

Now Brillig's *Extended Features* enables you to capture just about anything into an *Apple Writer* file for easy processing and to do all sorts of things with those or any other *Apple Writer* files. In addition, these utilities make it possible to produce multiple copies of *Apple Writer* files and to insert variables in those copies; one use of this feature is the printing of personalized form letters—from *Apple Writer.*

If you write programs rather than form letters, this package's utility for capturing Applesoft programs on the word processor is worth the price of the whole package. Once your program is in *Apple Writer,* you can edit single characters instead of full lines, change variable names, find specific GOTOs and GOSUBs or all of them in context. When you're satisfied, a simple procedure converts your file back to a program.

One of the features of *Extended Features* is the ease of using them. Some procedures require several programs to complete; Brillig has provided exec files that handle all these details. When you want to convert one kind of file to another, you just make sure your choosing the correct conversion and say EXEC that procedure; the program takes care of the manipulations after that.

Specifically, *Apple Writer Extended Features* contains programs: to convert text files to *Apple Writer* files and *Apple Writer* files (binary) to text files; to convert Applesoft programs to *Apple Writer* files and vice versa; to edit a DOS text file to easily create and change exec files; and to print the text file produced by the print time exit procedure.

In addition, a print time exit provides the ability to underscore and use a bar in *Apple Writer;* The generation of hex characters for enhanced mode printing; the ability to send all printed output to disk instead of to the printer; and the ability to override the fill justify mode of *Apple Writer* when you like.

As a bonus, Brillig has included a program to intercept the reset key with autostart ROM.

The convenience of the facilities provided by *Apple Writer* with the *Apple Writer Extended Features* might well justify the purchase of *Apple Writer* itself, if you haven't already got it, just to be able to use these utilities. MCT

Apple Writer Extended Features by Paul Malachowski and Kevin Cooper, Brillig Systems (Burke, VA). 48K, Apple Writer. $29.95.

The Warp Factor. By Paul Murray. Patience is not only a virtue, it's a requirement for playing a game with the depth of *The Warp Factor.*

Already flying high on the Top Thirty charts because of the inherent interest Apple owners have in anything dealing with space and the future, the program appears to have the kind of universal appeal and playing depth that could result in it becoming one of the all-time bestsellers.

But patience it does require. This is no aim-and-shoot game where all the skill and knowledge you'll ever need can be acquired in the first five minutes with the program. This is truly a game of strategy and tactics—in their truest and finest sense—and it will take multiple playings to gain rudimentary skill and understanding. Clearly that dictates against purchase by those who don't have the patience to master a complicated game. For those who do, *Warp Factor* promises untold hours of enjoyment as their proficiency grows.

Author Paul Murray has introduced sufficient variables into the program that no two encounters will ever be so similar as to invite ennui. The program contains information on the space ships of six nations. Twelve different ships are provided for, each with their own unique weaponry, staffing levels, and maneuvering ability.

The genius of *Warp Factor* is that each player chooses his nationality and, within the bounds of the weaponry available to that nationality, his fleet.

What this leads to is ever more complicated scenarios as the player's skill increases, but the difficulty factor can be controlled by the player himself to suit his competency. Will it be one Reman destroyer or five Klargon dreadnoughts?

The program basically provides for two modes of play—solitaire or two player. In the solitaire mode, you choose your own force and the Apple's force. Be apprised that Apple is a formidable opponent. In the two-player mode, each player chooses his own fleet.

At the end of the *Starship Operations Manual* that accompanies the game is a tip on how to convert the game to handle as many as eight players. Eight equally capable players in a free-for-all makes for as dynamic and interesting combat as has been witnessed on the Apple.

Software reviewers have found to their sorrow that it is never appropriate to proclaim any program as the epitome of the programmers' art for a particular genre. For as certainly as such rash proclamations are made, the very next software release will represent a quantum leap in the art.

But no matter how cleverly some subsequent programmer makes a space battle game, it seems unlikely that the challenge and diversity of *Warp Factor* will soon pale. ART

The Warp Factor by Paul Murray, Strategic Simulations, Mountain View, CA. 48K. $39.95.

The Complete Graphics System. By Mark Pelczarski. It's hard to write about a program like this one, because there are so many different things it will do. *The Complete Graphics System* lets you draw in two dimensions, assemble two-dimensional panels into three-dimensional figures, manipulate your three-dimensional figures in all sorts of ways, design hi-res text fonts and incorporate text into hi-res pictures, create shape tables, fill drawings with standard and composite hi-res colors—and more. The program earns its name, because it brings together at a modest price so many different graphics tools.

Two-dimensional drawings are made with the game paddles, using a system of fixed and movable cursors. You set one cursor at the beginning of a desired line, move the other to the end of the line, and hit the paddle button. Or you can just hold the button down while you move the movable cursor. Besides this point-to-point method, there's an option that allows you to draw a lot of lines radiating from a single point. In addition to straight lines, you can create circles, ellipses, or portions of either. The program offers the six Apple hi-res colors for lines, but the autofill routine will shade enclosed spaces with any of 108 composite hues. A palette feature shows all the colors at your disposal.

For more painterly effects, the system offers a set of nine paintbrushes. Just set brush to canvas with a paddle button and turn the dial. The various brushes yield different stroke widths and textures.

As you draw, the program displays available commands and the x and y coordinates of your movable cursor in a text window at the bottom of the screen. Pressing escape makes the text disappear, allowing you to use the full hi-res screen for your picture.

Unfortunately, the system lacks a convenient way to erase. If you can find your way back to the exact end points of a line, you can try redrawing it in black, but results are unpredictable because of the way the Apple plots hi-res colors. You may just wind up with a dotted line. If your drawing is complex, the safest approach is to prepare for error by saving frequently.

The text module comes with two somewhat antique-looking default fonts, one large, one small. With these you can add text to your pictures, in destructive, nondestructive, or reverse modes. The large font is available in all 108 colors, although the documentation warns that some colors will not produce acceptable resolution.

You can edit these fonts or create and save entirely new ones in either of the two sizes—a seven-by-eight grid for the smaller and a fourteen-by-sixteen for the larger. The fonts also include control and escape characters, so you can design special symbols or logos to go with your alphabets.

A panel module lets you create three-dimensional figures by drawing two-dimensional faces one at a time and joining them. Once you've got a three-dimensional object assembled, you can manipulate it with the system's 3-D graphics module. Here your options include rotation on any of three axes and about any center you choose; linear movement of the entire figure, expansion and contraction from any center point, and distortion. The last of these choices, distortion, is in effect a one-dimensional scaling operation in the direction of height, width, or depth.

Additional features of the system are a shape module, which allows you to create shape tables either with paddles or with keyboard, and a shrink utility, which reduces three-dimensional objects by factors of four and puts them into any of the four quadrants of the screen.

The Complete Graphics System will take a little time to master. Fortunately, the author has provided a thorough and clearly written manual. Besides describing every aspect of the program, the manual explains how to use pictures, shape tables, font, and the 108-color routine in your own programs. Another nice touch in the manual is a complete outline of the command structure, listing every menu and submenu in the system.

The Complete Graphics System is available either as a single package, or as two separate items. The smaller units are the *100-Color Drawing System*, containing the drawing, text, shape, and shrink modules of the complete system, and the *3-D Drawing System*, which contains the 3-D and 3-D panel modules.

The Complete Graphics System, by Mark Pelczarski, Co-op Software, West Chicago, IL. 48K, Applesoft. $59.95. *3-D Drawing System* and the *100-Color Drawing System* same requirements, each $32.95.

Beneath Apple DOS. (Book review.) By Don Worth and Pieter Lechner. With incredible hardware and software appearing daily, it's hard to get excited about a new book, even one specifically written about the Apple. However, Worth and Lechner unveil so many of the mysteries of Apple's Disk Operating Systems that you may find it difficult to put the book down before finishing it. In fact, so much information is crammed into this 160-page spiral-bound manual that it could have been titled *Everything You Ever Wanted to Know About Dos (But Apple Didn't Tell You)*.

Beneath Apple DOS serves two purposes. First, it provides a general description of the entire disk system, including how data is actually stored on the diskette. This is invaluable if you're trying to recover a clobbered disk. Machine language access to DOS is fully covered with an entire chapter devoted to detailed analyses of Read/Write Track/Sector (RWTS) and the File Manager. Exactly how much you ingest on the first reading will depend on your prior knowledge about the workings of DOS.

The book's second function as a comprehensive reference manual ensures it a prominent place on many bookshelves. *Beneath Apple DOS* includes a tear-out reference card much like that found in the Applesoft II and Apple DOS manuals.

Although this book is by no means aimed at the neophyte, it is very well written in clear, easy language augmented with numerous charts and figures. There's even a sprinkling of cartoons to illustrate some of the major points, and there's a complete glossary of disk-related terms.

A real plus is in Appendix A: five assembly language disk utility programs. The source code is given (a valuable learning aid itself), but all programs can be entered from the Apple's monitor. Another appendix describes some of the earlier disk protection schemes. In this area, the book only scratches the surface; those with ulterior motives are not served.

Beneath Apple DOS begins with a brief history of DOS (3, 3.1, 3.2, 3.2.1, 3.3), pointing out most of the changes made at each phase. Next, diskette formatting is covered for both thirteen-sector and sixteen-sector types, describing exactly how data is nibblized for storage on the disk.

Although this is one of the most complicated and misunderstood concepts about DOS, the authors have done an excellent job explaining this process in detail. Organization of the diskette is then presented, giving treatment to the VTOC (volume table of contents), catalog, and track/sector lists. The precise formats of Basic, binary, and text files are also depicted. With this information, you should be able to repair many of the typical problems that can make a disk unreadable.

The book continues with a detailed look at the DOS program itself, including the booting process and zero page usage. Here, one will find the real meat of the subject, annotated to the DOS on almost a byte-by-byte basis. Programmers attempting to modify DOS for their own uses will appreciate the mammoth amount of work that went into this section. For copyright reasons, a disassembled listing of DOS is not included.

Beneath Apple DOS is not for all Apple users. For example, there's nothing on Basic programming; the Apple DOS manual covers this. But if you're writing machine language programs that interface with the disk or customizing DOS for special uses, this book is a must. JH

Beneath Apple DOS by Don Worth and Pieter Lechner, Quality Software (Reseda, CA). Paperback. $19.95.

Battle Cruiser Action. By Frank Heffner and Bob Reynolds. In the tradition of warfare, it's perhaps fitting that an issue featuring the first wargame publisher for the Apple, Strategic Simulations, should also mark the debut to these review pages of what may well be SSI's first major competitor. Mytopia Gameware Institute was also founded by people who were both avid war gamers and computer enthusiasts; there the resemblance ends. Where SSI strove for immediate production of large-scale, elaborately packaged, fully equipped translations of their favorite games to the Apple, MGI chose the long-range approach to create an original war game for the Apple, spending three years in meticulous research to ensure historical accuracy and in careful program planning to achieve simplicity of game mechanics.

Says game designer Frank Heffner in the notes to *Battle Cruiser Action*, "We wanted a game that featured naval maneuver in its purest form; one that eliminated outside paperwork and calculations so that we could spend more time enjoying the game."

So this war game has no paraphernalia, save its manual and disk; it comes in a modest package, the manual cover serving as the display. On the screen too, elaborate graphics have been scorned in favor of straightforward tactical space and memory room for historical strategic detail. Each ship is carefully defined with the specific characteristics it actually had in terms of armor thickness and positioning, structural integrity, crew quality, damage control ability, gunnery ranges, shell effectiveness, and speed. All these are taken into account in the three characteristics the player must be concerned with: speed factor, turret factor, and defense factor.

Once the ships were defined, Mytopia concerned itself with the player: removing the necessity for mechanical aids such as graph paper, reducing the time needed to enter data while retaining the maximum possible player control of ships.

Finally, the focus became making the game challenging to everyone, regardless of expertise or lack of it. Six levels were built in to the game; levels are accomplished primarily through varying the ability on the part of the German admiral and through differing German gunnery effectiveness and aggressiveness. The novice will shortly be able to beat level one; the authors still can beat level six only 50 percent of the time.

The entire process took Mytopia three years. Upon first booting *Battle Cruiser Action*, hearing its one-line melodic call to arms and seeing its utile no-res graphics, the war gamer sated on colorful hi-res and animation, on elaborate charts and maps, may wonder where the three years went. But playing the game tells the tale. Tactics are all. You can try the tactics actually used by the British with these very ships in World War I, then try other approaches you invent;

you can try the methods of different generals, seeing if they would have done better. The tactical options involve the assigning of squadrons, arranging ships in effective fighting order, deciding whether to take positions broadside, head-on, or moving away in respect to the Germans, anticipating the Germans' moves. How you direct your ships, individually or in squadrons, in battle will determine the outcome. Chance plays a minor role. The true tactician could not ask for more.

There is a purity and an honesty to *Battle Cruiser Action* that is refreshing among the razzle-dazzle and gimickry. For the serious war gamer, it's the real thing. MCT

Battle Cruiser Action by Frank Heffner and Bob Reynolds, Mytopia Gameware Institute (Sioux City, IA). 48K. $39.95.

Six Micro-Stories and His Majesty's Ship "Impetuous." By Robert Lafore. The name of the publisher—Interactive Fiction—heralds a new fantasy genre for the Apple. These stories are like adventure games in that they require you to make decisions that control your fate. But they are more literary than gamelike. The pleasure they offer is akin to that afforded by an engaging, well-written novel. Naturally, you're the protagonist. In *"Impetuous,"* you captain a British ship of the line in her battles with the Spanish and the French. In every chapter there are tactical and moral dilemmas to be solved. Do you engage the enemy or not? Do you dare defy orders for the sake of chivalry? Are your own subordinates plotting against you? You make the fatal decisions by entering your end of the dialogue. Decisions have consequences, of course, some of which you may not see until you are five chapters or more down the pike.

The *Six Micro-Stories* are little vignettes that suggest, more than fulfill, the possibilities of the form. In one, you meet the lady of your dreams in Golden Gate Park. Can you persuade her to spend some time with you? In another story, you are caught trying to return a piece of jewelry stolen by someone else. Can you explain your way out of this unlikely predicament?

Part of the appeal of both *"Impetuous"* and the *Micro-Stories* is that they are nicely written. The language is a pleasure to read. And while the author suggests that you keep your dialogue fairly straight and pithy, the programs will understand and respond to most of what you're likely to say. CS

His Majesty's Ship "Impetuous" and *Six Micro-Stories* by Robert Lafore, Interactive Fiction, Inverness, CA. 48K, Applesoft. *"Impetuous,"* $19.95; *Six Micro-Stories*, $14.95.

Real Estate Analyzer. By Dr. James Howard. Howard Software Services claims to publish software for the serious user; in the *Real Estate Analyzer* they aren't just whistling Dixie.

The *Real Estate Analyzer* is so powerful that anyone less than a serious investor in income properties will believe the program a bomb because the data it generates oftentimes belies common sense.

Dr. James Howard has packed so many esoteric, albeit important, variables into 48K that only the investing cognoscenti will recognize the worth of the program and its data.

Analyzer is reminiscent of *VisiCalc* in its ability to answer the question "What if?" The difference is that this is a dedicated, special purpose piece of software.

While it's conceivable that a home buyer might find the program useful, the strength of the software, for the most part, is in dealing with relatively conventional income property transactions.

The program will definitely not handle the kind of transactions so euphemistically called creative financing that allow would-be home owners to buy with unconventional loans. And that's as it should be. Anyone who needs creative financing instead of a conventional mortgage to buy a home is not in the market for income property in the first place; and it's income property transactions to which the program is dedicated.

To say that *Analyzer* doesn't handle creative financing situations is not to indicate that it isn't versatile in almost every facet of conventional financing.

You can conjure up a scenario that calls for a forty-thousand-dollar first mortgage at 15 percent being paid on an amortization basis, a two-hundred-thousand-dollar second mortgage at 19 percent on which you pay interest only, and a seventy-thousand-dollar third mortgage to your brother-in-law at 12 percent on which you only pay two thousand dollars a year.

You then plug in all manner of variables: closing costs of the purchase, several common operating expenses that are then totaled, assumptions about inflation, the rate of increase of property values, the rate of change of purchasing power of the dollar, your tax bracket for income, your tax bracket for capital gains, the income you expect from the property, ad infinitum.

Then the program goes to work. It will do cash flow analyses to determine if the project will be self-sufficient. It will also do six different analyses of investment return.

If you don't like the results, go back to the beginning and plug in different assumptions. If the project looks good to you, find two bankers and a brother-in-law who agree.

The strength of the program is that it takes into account almost every penny that will be spent or earned and even calculates the tax consequences should you decide to sell.

The weakness of the program lies in the old saw: "Figures never lie, but liars figure." You can make an investment look really enticing by entering assumptions that are favorable—even if they are also unrealistic.

If you are not sophisticated enough to know that you've seeded the program with favorable data, you could close a deal that will send you to the poorhouse.

On the other hand, if you can provide the program with a realistic set of parameters within which to work, you could soon be rolling in green stuff. One thing's for sure: if this program can't enhance your investment chances, you're not as smart as you thought you were.

The software includes a versatile print report module that allows you to print results of analyses or to print the assumptions you originally fed the computer.

The documentation includes some brief appendices that will help the novice by defining terms and giving a brief discussion of the technical basis of investment analysis. Also included is a chapter on making investment decisions. These concise explanations should take some of the mystery out of the program for those who come to it green. ART

Real Estate Analyzer by Dr. James Howard, Howard Software Services, La Jolla, CA. 48K, DOS 3.2 $150.

Pulsar II. By Nasir. What you've seen in the arcades as Star Castle has now been adapted to the Apple in *Pulsar II*.

The problem with that lies in the original game itself. Star Castle may be the second most overrated of all the arcade games, made popular mainly by virtue of its luminescence. Nevertheless, *Pulsar II* is an interesting diversion.

Your ship circles the shielded star, either clockwise or counterclockwise. From your ship, you shoot straight-pathed missiles at the shields. As in the original, hitting the shields causes an enemy missile to form and come out at you; after a time, its duration depending on the level, the missiles actively chase your ship.

There are some good innovations on this disk. *Pulsar II* is accompanied by a second game, *Worm Wall*. Scores of the two games can be maintained cumulatively, so that you can finish a level of one, move to the other for a level of it, then go back to the first game for the next level.

Worm Wall is an extremely frustrating game, but that's its intent. It consists of a simple maze of concentric areas with moving half-openings in the walls. You travel through one layer of the maze, waiting for two half-openings to meet and form an opening you can jump through. Better be in the right spot at the time, though. This is the only way to get to the next layer.

Oh, yes, there's the matter of the hungry little worm heads. There's one in each layer, and when it hears you jump in, it immediately starts after you in hopes of supper. If it catches up with you or you run into it, you're the supper. And if you don't get through to the next level smartly, the worm, as is the way of worms, becomes two worms, proceeding after you at different speeds.

On the other hand, if you make it to the middle circle, you win the round (no, you don't have to eat the worms).

Both programs are in colorful hi-res animated graphics; but the fine detail and rich color of other Nasir works are missing. ART

Pulsar II (with *Worm Wall*) by Nasir, Sirius Software (Sacramento, CA). 48K, $29.95.

Orbitron. By Eric Knopp. Eric Knopp has a fine instructor at his disposal, and that master's touch is recognizable in Knopp's first professional programming effort. We can see Nasir in the color and shapes and in the careful graphic design. But beyond these hints, *Orbitron* is all Knopp's. The student has learned well, and he has made the knowledge his own.

Orbitron is a first-rate arcade game for the Apple. Aim and timing are the skills required. Born of an idea taken, not from the arcades, but directly from the middle-teenage imagination of Eric Knopp, *Orbitron* places you in the semi-shielded center of the screen. First, if you're quick, you might pick off one or two wandering asteroids for a very large score gain. If you're very quick.

More likely, you won't even realize the game's begun until a warning flashes on the screen, with appropriate sound effects. The warning is against seven enemy orbs that zoom into an arc formation above you. As they begin building shields, they take turns sending out missiles to get you. The missiles orbit you about once before they attack, and it's wise to get them during the orbit; when they do attack, your shield is ineffective.

Orbs score more than the missiles, but the missiles are the danger. Once each orb has fired a missile and the last missile has been exploded—either by you or on you—the rest of the orbs take off. Assuming you're still alive—you get five ships—when the orbs are either destroyed or gone, you score for each ship you have left and prepare for the next level.

But you're not there yet. First, two enemy space ships appear in succession, leaving one stationary but deadly missile each. You must draw a bead on the missile and take it out immediately; you only get three or four shots before it attacks. Having survived this ordeal, you are *really* promoted to another level, and are immediately beset by another siege of asteroids, another warning, seven more orbs, etc., etc.

If Nasir himself had authored *Orbitron*, it would be well-received, but the fact that it is the first effort of a teenager makes *Orbitron* very special indeed. Eric Knopp appears to be on the way to an exciting career. MCT

Orbitron by Eric Knopp, Sirius Software (Sacramento, CA). 48K, $29.95.

GOTO 64

THE PASCAL PATH

By Jim Merritt

Tools of the Craft, Part One

In my opinion, programming is a *craft*, requiring, as does any craft, the application of special skills and special tools to produce objects—programs—that are valuable at least for functionality, if not also for their aesthetic qualities. The beginner's first "toolkit" is simply a programming language, and, as any craftsman will tell you, it's impossible to master your craft without first mastering your tools. In the next installments, then, we'll study the features of Apple Pascal, paying close attention to the philosophical issues that motivated their inclusion in the language.

Knowing the "why" of a language construct allows you to use that construct most efficiently, wherever and whenever it can do the most good, and also helps you identify situations where its use would be inappropriate. For instance (to borrow from another craft), driving nails with the blunt end of a hatchet, although possible, is not something you would normally do, since a hammer is not only suited, but *intended*, for the job. My task for the next few months is to teach you to distinguish Pascal's "hammers" from its "hatchets." Once you are familiar with the tools in Pascal, you can apply them creatively and effectively to the solution of your own problems.

A computer program does three things: it defines certain *data*, specifies the *actions* that are to be performed on that data, and prescribes the sequence in which those actions must occur (the *control flow*). Let's examine the concepts of data, action, and control flow individually and cover a little Pascal syntax in the bargain.

The job of any program is to manipulate, transform, transmit, or receive one or more forms of data. The data can come from the outside world or can be contained within the program itself. The word *data* is the plural of *datum*, which we define as the representation of some unit of information. Both the representation and the information it signifies are entirely up to the programmer. Much of programming involves defining the information you want to process, then defining data structures to represent that information during the execution of a program.

Data Types. Pascal provides four different types of data values that you can use to represent your particular information. These are Pascal's fundamental data types, and their names are Char, Boolean, Integer, and Real. (A fifth fundamental type, Pointer, is also available; its underlying philosophy and use will be the subject of a future column.)

Char. The Char type consists, in Apple Pascal, of all the ASCII characters, including both the upper-case and lower-case Roman alphabets, the ten digits 0 through 9, several punctuation marks, and certain invisible characters that are used to control I/O devices (for example, to tell a printer to eject a page of paper). A datum value of the Char type is exactly one character.

For those who haven't yet learned, ASCII is a standard, ordered character set, in which a certain character value is mapped onto a whole number in the range of 0 to 127 inclusive. The character *A*, for example, corresponds to the number 65. The lower-case version, *a*, corresponds to 97.

Take a look right now at the complete listing of the ASCII character codes, given in Table 7 of the *Apple Pascal Language Reference Manual.* Note that the characters corresponding to the numbers 0 through 31, and also 127, are invisible control characters. Should such a character be sent to an output device, you are not likely to see displayed any of the two- or three-character names given for these codes. The names are provided in the table only to help you identify the standard special function of the character. (We'll talk about some of these in the months to come.) The character corresponding to 32 is a blank, or space, and that's what you'll see (not the name "SP") if you send it to an output device. All the other characters correspond to the letters, digits, or punctuation marks shown in the table.

Boolean. The Boolean type is very small, consisting only of the values *true* and *false.* These two values, however, are ubiquitous in Pascal, and so we'll be speaking of Boolean data quite often.

Integers and Reals. Two of the four fundamental types, Integer and Real, are numeric in nature. An Integer number is a whole number (i.e., no fractional part) and, in Apple Pascal, may range from −32768 to +32767. Real numbers include both a whole part and a fractional part. Numbers such as 3.33333 and −1.414 are examples of Pascal's Real numbers. People seldom differentiate between Integers and Reals in daily life—numbers are numbers. Pascal includes the distinction because numbers are used quite often in programming, and it turns out that the representation of Integers at the level of the computer chip itself can be made much simpler and more compact than that for Reals. Computations with Real numbers also tend to take longer for the machine to accomplish than Integer arithmetic. (Because the machine is so much faster than you anyway, you may never be aware of the difference in computing speed between the two types of numbers, unless you compare the performance of the program that does thousands of Integer operations with the performance of one that does the same amount of similar Real operations.)

Literals. There are two ways to refer to datum values in Pascal programs; explicitly or by name. If you wanted to use the number 10 in the program, you could simply include the number itself at the appropriate point. An example of this would be the statement

```
WriteLn(10);
```

which would cause the number 10 to be written out onto the system's primary output device (the *console*). This illustrates the use of a *literal constant* (abbreviated as *literal*). You may form and use literals for the values of any of the four fundamental data types. Figure 1 includes the syntax graphs for each kind of literal, and you should try following different paths through each of the diagrams to get a feel for the proper way to form the corresponding literal. If you do, you should notice certain facts:

1. Char literals must begin and end with an apostrophe ('), and contain exactly one character between the two apostrophes. A special case is when you wish to denote the apostrophe itself. Then, you use two consecutive apostrophes, so the resulting literal consists of four apostrophes in a row ('''')!

2. Boolean literals must not be surrounded by apostrophes or any other characters. A Boolean literal value is written in programs as is. The syntax diagram doesn't tell you, but *true* and *false* are actually identifiers, so Pascal doesn't care whether you write them in upper case, lower case, or mixed case.

3. Real literals must contain a decimal point and include at least one digit to the left and one digit to the right of the point. A real number is actually made up of two parts. The part that contains the decimal point is the numeric value itself and is called the mantissa. The second is an optional decimal scaling factor, called the exponent. Whenever an exponent is present in a Real literal, Pascal multiplies or divides the mantissa by the power of ten indicated by the exponent, in order to get the actual value represented by the literal. Whether it multiplies or divides is determined by the sign of the exponent (not the sign of the mantissa!). An exponent of "e2" means "multiply the mantissa by 10 to the second power, or 100." An exponent of "e−2" means "divide the mantissa by 10 to the second power." If the exponent part is omitted in a Real literal, it is assumed to be "e0," that is, 10 to the zero power, or 1. The syntax diagram doesn't mention it, but the "e" in the exponent part of a Real literal may be upper case or lower case. Also, exponents may range from e−38 to e+38.

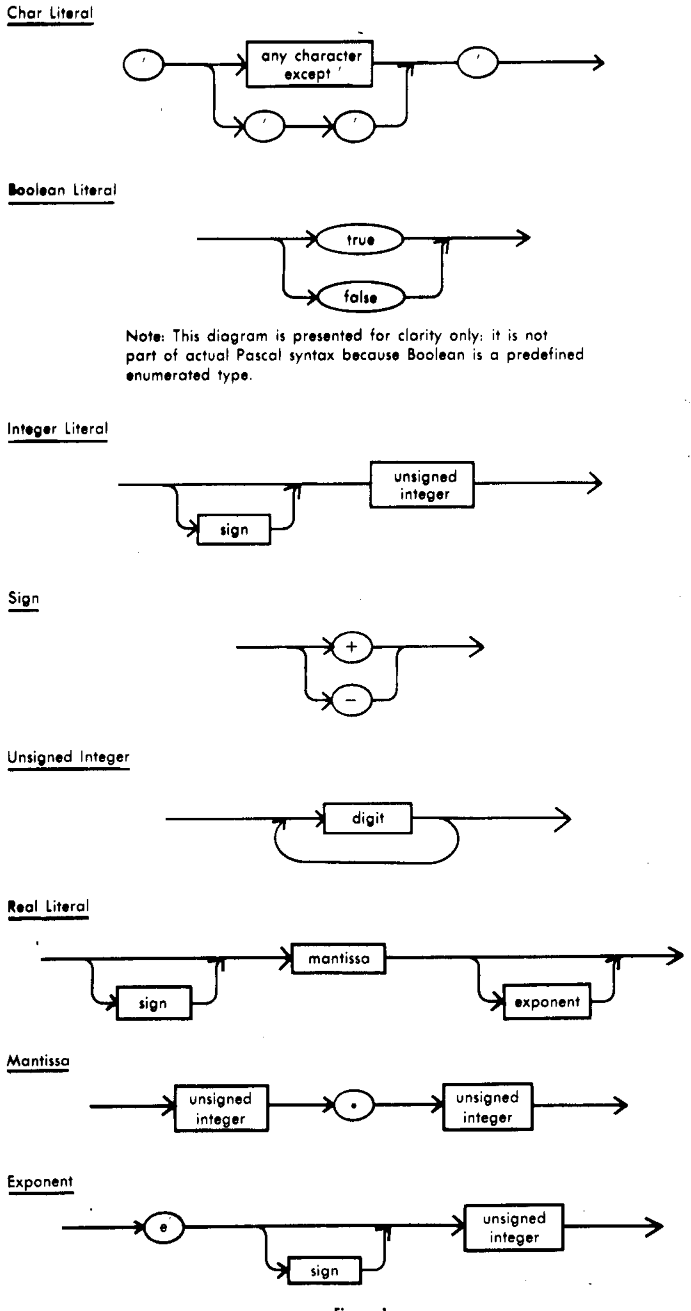


Figure 1.

Here are some legal literals, followed by some illegal ones. Each of the four fundamental types is represented. For each example, trace through the appropriate syntax diagram to convince yourself of the literal's correctness or incorrectness.

CHAR					
Correct:	'a'	'H'	'*'	' '	'8'
Incorrect:	a	''	'HH'		
BOOLEAN					
Correct:	False	True	FALSE	TRUE	
Incorrect:	F	T	'True'		
INTEGER					
Correct:	−0	+0	0	32767	45
Incorrect:	33000	62.	−.3	1,024	
REAL					
Correct:	0.0	−1.23	0.3		
	7.5E−1	2.0e+7			
Incorrect:	.3	1	−46.	4.5e100	

Named Constants—the CONST Section. Suppose you must use a certain number, say pi, in many different parts of your program. Pi is roughly 3.14159. You could sprinkle this number around your program as a literal, but you might transpose a digit or two here and there, and that would play interesting (and frustrating) tricks with your program's accuracy. Moreover, weeks or months after writing the program, you might decide that you want to use another number instead of pi. Trying to locate and change all occurrences of 3.14159 to occurrences of the new number could be a major job if your program is long and the number occurs often.

Pascal permits you to associate meaningful names with arbitrary constants, and you can take advantage of this feature to solve both of these hypothetical problems. Constant names are formed according to the rules for identifiers that you have already learned if you have been following the Pascal Path from month to month. Otherwise, refer to Appendix F of the *Apple Pascal Language Reference Manual.* Equating names with constant values is done in the CONST section of the *declaration area.*

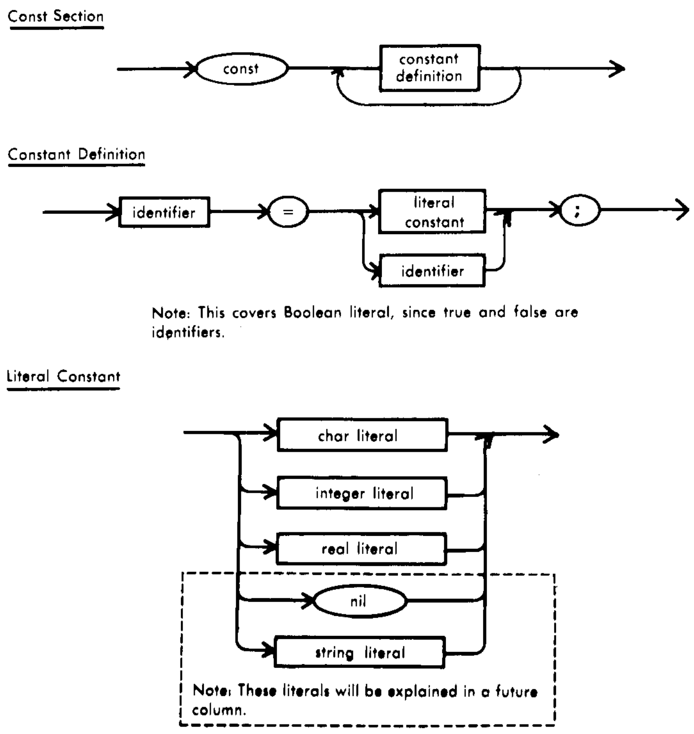


Figure 2.

The declaration area is that portion of your program which lies between the program heading and the BEGIN keyword that starts the main body of the program. In it, you may declare (define) any objects your program needs that aren't already provided to you by the Pascal language. The program we compiled a couple of months ago, *SomeExpressions*, used only things (such as "WriteLn") that Pascal gives to every programmer. Its declaration area was, therefore, empty.

The optional CONST section is, in most programs, the first in the declaration area. (It may be second, but we won't cover this situation until much later.) It consists of the keyword CONST, followed by as many constant definitions as you wish to list. Figure 2 gives the syntax diagrams associated with the CONST section. In each constant definition, an identifier name is equated to a constant value. That value may be expressed by a literal, or by an identifier that names a previously defined constant. Unless you take special action to the contrary, such a definition will hold throughout the rest of the program, and anywhere you use a constant name, it will be as if you wrote the corresponding literal.

Now, we can give a name to the Real constant 3.14159:

```
PROGRAM
  Dummy;
  CONST
  Pi=              3.14159;
  MagicNumber= Pi;
BEGIN
END.
```

Note that the only reason I framed the CONST section within a dummy program was to depict graphically the placement of that section within the declaration area.

I have defined two constants here to solve the two software development problems mentioned earlier. The compiler can't detect digit transpositions in numeric literals—one number looks like any other number to it. Use of the constant identifier "Pi" throughout the program, instead of the literal "3.14159," will ensure that the same constant value is referenced everywhere and that no inadvertent digit transpositions due to typographical error occur. You might, of course, misspell the identifier, but if you do the compiler will probably be able to inform you of your error, because, chances are, no other declared identifier will be the same as your misspelling of "Pi." So, the compiler will assume that you are trying to use an undeclared identifier (a name that hasn't been associated with any object). This is a syntax error condition (#104), and the compiler issues an error message whenever it occurs.

Of course, should you misspell "Pi" as, for instance, "Pii," "Po" or "Pu," and any of these alternate spellings corresponds to another Real-number constant declared within your program, the compiler will use the wrong constant, not the one you really mean. Therefore, you should be careful to keep the spellings of your identifiers as distinct as possible so the compiler can be of most help to you in detecting misspellings.

In the second problem area, I associated the identifier "MagicNum" with "Pi." Throughout your program, there may be places where the use of the number represented by "Pi" is appropriate, but where other numbers could be used, as well, to achieve different results. Going in, after the program has been written, and changing all occurrences of the Literal "3.14159," or even the identifier "Pi," to something else, is boring, error-prone work.

When writing your programs, you should relax and give yourself plenty of time to consider whether or not you will ever have occasion, during the normal evolution of the program, to change certain parts of it. Those parts should be written to permit easy change, whenever possible. In this case, defining "MagicNum" as equal to "Pi," then using its name instead of "Pi" wherever you feel you might want to use a different number later, will make it possible to switch to that new number simply by updating the single definition of "MagicNum." In doing so, you will still have "Pi" around to use as you see fit.

I realize that anticipation of the normal evolution of a program requires a reasonable amount of programming experience on your part, and I don't expect you to be able to plan for every contingency when writing software. Even proven, seasoned programmers consider themselves fortunate when it appears as if they have covered all bases with a given program. Nevertheless, the sooner you accept that change (in the form of updates, bug fixes, and so forth) is a routine part of software development, the sooner you will incorporate into your personal methodology techniques that allow for change, thus easing your overall programming burden.

Your Own Types—the TYPE Section. One of the things that has helped give programming its unfortunate reputation as an arcane art is the fact that, typically, a programmer must take a clear, English language description of a process or program solution and reexpress it in specific computer terms of programming language. The resulting program bears vague resemblance to the original description, and this is primarily because of the necessary transformation of the program's data descriptions into machine-acceptable ones. For example, suppose your program must process many yes and no decisions. The information that must be represented consists of the concepts *yes* and *no,* but no programming language, not even Pascal, gives you a data type that includes those values. In most languages you must force data values from other types to act as your yes and no. In Pascal, for instance, the following CONST section would establish two new named constants, "Yes" and "No," as integer values:

```
CONST
  Yes= 1;
   No= 0;
```

Or, the Boolean type could be used:

```
CONST
  Yes= True;
   No= False;
```

Finally, yes and no might be represented by values of type Char:

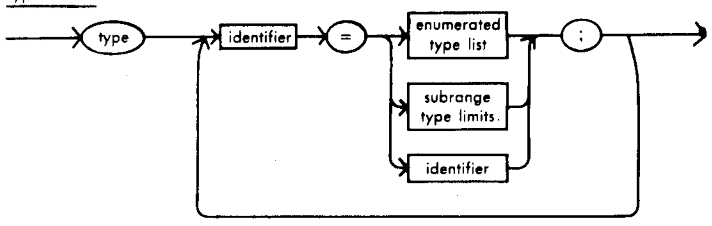
```
CONST
  Yes= 'Y';
   No= 'N';
```

There is nothing sacred about the specific values I chose to associate with the identifiers "Yes" and "No" in each case. In the Integer example, "Yes" could have been equated to 43 and "No" to −711. In the Boolean case, "Yes" could have been false, and "No" true. With Char, I could just as easily have used lower case as upper case, or even different characters altogether. "Yes" and "No" could also be represented by Real numbers, but I'll spare you an example because I think you get the idea.

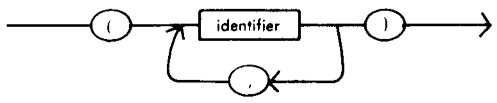
In a language like Pascal, in which it's possible to associate meaningful names to literal constants, it's quite easy to give new meanings to existing data values. In other languages—ones that require using the literals themselves instead of symbolic names—programs might lose clarity when this type of data aliasing is employed. For instance, try reading a program in which the programmer used 1 to represent yes, and used 0 to represent no. This can be doubly confusing if 1s and 0s standing for yes and no have been included in arithmetic formulas with 1s and 0s that are actually being used as regular numbers!

To avoid confusion with other forms of data, you might like to define entirely new data values corresponding to the information concepts yes and no. In effect, you would create a new data type, containing just those values. In Pascal, you are free to create your own custom data type, with a *type declaration.* Figure 3 gives partial syntax diagrams for the type declaration section. This section, like that for defining constants, is optional. If used, it is placed after any constant declaration section and before any other declaration section. The syntax diagrams for this section are incomplete, illustrating only those constructs we're now covering. The diagram, and this discussion, will be concluded further down the Path. We'll talk about two flavors of programmer-defined types this time: *enumerated types* (also known as "scalars"), and *subranges.*

Type Section



Enumerated Type List



Subrange Type Limits

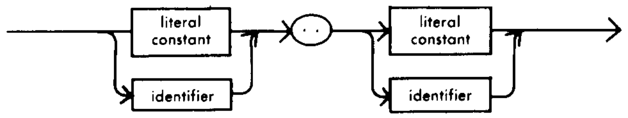


Figure 3.

Enumerated Types. A programmer-defined type containing the values yes and no is an example of an enumerated type. Here is the TYPE section that defines such a data type:

```
TYPE
  YesOrNo=
    (No, Yes);
```

From looking at both the example and the syntax diagrams, you can see that the declaration for a type is similar to that for a constant. One identifier is used to name the type itself, and other identifiers are given as the type's individual data values. The list of values is enclosed in parentheses, and each element is separated from its successor by a comma.

An important property of an enumerated type is that there is a definite ordering of its constituent data values. In the example type "YesOrNo," the value "No" is taken by Pascal to be "less than" the value "Yes," because it comes before "Yes" in the value list. There is no compelling reason why "No" should come first, in this instance. In many programs, however, the order of data values in enumerated types does make a difference and can make programming more convenient, as you'll see in future examples.

Enumerated types are just one way that Pascal lets you shape your program in terms of the problem at hand, rather than the other way around. If you need a specific range of values for a certain purpose and would prefer not to make existing data values do double-duty, you can make up a completely new type and use it as any other Pascal data type. Here are some more enumerated types, presented primarily to stimulate your imagination:

```
    TYPE
      Class=
        (Freshman, Sophomore, Junior, Senior, Graduate);
      TrafficLightState=
        (Go, OKLeftTurn, OKRightTurn, Caution,
FlashCaution, Stop, FlashStop);
      Color=
        (Red, Orange, Yellow, Green, Blue, Violet);
      Fruits=
        (Apple, Banana, Boysenberry, Cherry, Grape,
        Grapefruit, Lemon, Lime, Orange, Pineapple,
        Strawberry);
      Months=
        (January, February, March, April, May, June, July,
         August, September, October, November, December);
      Sex=
        (Male, Female, Other);
      Direction=
        (North, South, East, West);
      SysComponent=
        (Editor, Compiler, Filer, Assembler, Linker,
        Librarian);
      CarMakes=
        (AlfaRomeo, AMC, Audi, BMW, Buick, Cadillac,
        Chevrolet, Chrysler, Datsun, Dodge, Ferrari, Fiat,
        Ford, Honda, Jaguar, Lincoln, Mazda,
        MercedesBenz, Mercury, Plymouth, Pontiac,
        Porsche, Renault, RollsRoyce, Toyota, Triumph,
        Subaru, Volvo, VW, Others);
      Boolean=
        (False, True);
```

Did you notice the inclusion of Boolean in the list above? It turns out that Boolean acts as if it were defined in just the way shown here. Pascal would have left it to the programmer to define Boolean for every program needing it, but so many of the language's other built-in features and statements require the Boolean type that Pascal gives it to you for reasons of convenience. Note that the name of every fundamental type is an identifier, not a keyword.

In Apple Pascal, enumerated types may be of any size, but the compiler must consume memory during the compilation to store the name of a type and the names of all its values. If the compiler runs out of memory before making note of all the values in a given type, your compilation will fail. (This shouldn't happen until much later.)

Subranges. You can also declare types to be subranges of other types. When a subrange type is created, the type from which it is taken is known as the base type. A subrange type is specified by giving a type name, a lower limit from the base type, and an upper limit from the same type. The limit values are separated by a Pascal-style ellipsis, which is only two dots instead of the three dots you may be used to seeing in English prose. The subrange type includes all values between and including the limit values.

Here are some subrange type declarations. Some use the fundamental types for base types, while others are based on the examples given above for enumerated types. Your easy job is to identify the base type for each subrange.

```
      Summer=
        June . . August;
      UpperCase=
        'A' . . 'Z';
      StndSex=
        Male . . Female;
      Undergraduate=
        Freshman . .  Senior;
      Naturals=
        1 . . 32767;
```

Note that it is impossible to create subranges containing noncontiguous elements. That is, a subrange must consist of an unbroken sequence drawn from the base type. To illustrate, you couldn't create "Primaries," a subrange of "Colors," containing only the values "Red," "Yellow," and "Blue." Nor, from the Integers, can you draw "Primes," containing only the prime numbers. Neither of these proposed subranges is contiguous. There is, however, a Pascal construct known as a SET that is provided specifically for the manipulation of such noncontiguous groups. We'll talk about SETs soon.

I am slightly embarrassed that, at this stage of the game, I can't give you a truly convincing reason for the inclusion of subranges in Pascal. We will have to go a little farther before their true utility becomes apparent to you. In fact, we'll travel most of the way in the next installment, so I hope you'll be back next month, as we take up the concepts of variables, operators, expressions, and assignment.

Beginners' Corner

By Craig Stinson

Beginners' Corner began last month with an introduction to some of the programs on the System Master disk. In particular, we looked at *Little Brick Out, Applevision, Brian's Theme, Color Demo,* and *Color Demosoft.*

There are a couple of other interesting programs on the System Master that we didn't get to last month. In one—a game called *Animals,* written in Integer Basic—the computer asks you questions and tries to guess the animal you're thinking of. What makes this game different is that the computer learns as it loses; every time you beat the program, you teach it something new about the animal world.

The other goody is a program called *Phone List,* written in Applesoft, which allows you to create your own personal, computerized, little black book. You can store up to a hundred

Glossary

Backup—A duplicate of valuable programs or other data.
Default—An answer that the computer presumes to be correct unless informed otherwise. Defaults serve a dual function: they save keystrokes and help the user avoid erroneous input.
Disk controller card—A circuit board that interfaces the Apple to one or two disk drives.
Read—To retrieve data from a disk.
Ribbon cable—A flat, multistranded connector.
Write—To store data on a disk.
Write-protect—To prevent the computer from storing data on a disk. This is accomplished by putting a piece of tape or a special write-protect tab over the notch near the upper right-hand corner of the disk.

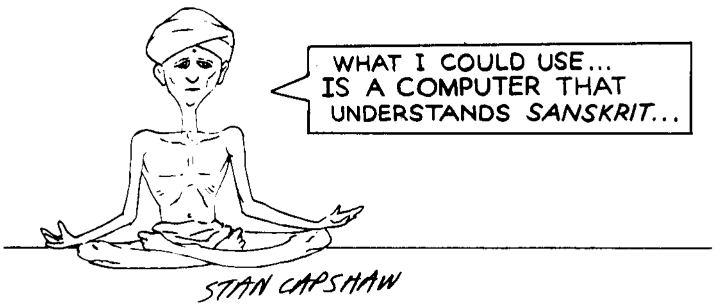
names and numbers. Later, when you want to recall a number, you can do so by typing as few as two adjacent letters of the name.

Before you can use either *Animals* or *Phone List*, however, you will need to copy them from the System Master onto another disk. That's because the System Master is a write-protected disk.

Our activities last month all required the Apple to get information from the System Master disk. This process is known as reading the disk. Nothing we did, however, required the computer to put material back, or *write*, on the disk.

But if you're going to teach your Apple about animals or ask it to memorize the names and phone numbers of your hundred closest associates, you're going to have to give it a place to store its new information. So this month we'll be writing as well as reading.

To begin, you will need at least one blank disk. Apple supplied you with a blank disk when you bought your system. If



you still have that one, we'll use it now. Otherwise, any other blank disk will do fine.

Compare the physical appearance of your blank disk with the System Master disk. You'll notice one difference—a little notch on the blank disk, about an inch away from the end that bears the label. That notch makes it possible for the computer to write on your disk. If the notch is open, the disk is said to be write-enabled; if it is covered or nonexistent, the disk is write-inhibited or write-protected.

Keeping Tabs on Your Disks. Apple made the System Master a write-protected disk so that you wouldn't accidentally overwrite or alter any of the crucial programs it contains. Later, when you have vital data of your own on disks, you can protect it the same way by putting a piece of tape—or a special write-protect tab, usually supplied with blank disks—over the little notch.

The recommended procedure is to copy your entire System Master disk to provide yourself with a backup in case the master disk should fail. It's unlikely, of course, but it is possible that someone will come along and put a cigarette burn in your disk, or bend it, or lose it. And even if nothing like that ever happens, you would still be wise to make a backup because disks have a finite lifetime—something on the order of forty hours of actual playing time (time when the disk drive's red light is on).

The System Master contains two programs that are used for copying entire disks. They are *Copy*, for machines using Integer Basic, and *CopyA*, for machines that use Applesoft. An additional program, *FID* (File Developer) will copy individual files or programs from one disk to another. *FID* is a machine language program that can be run by both Integer and Applesoft Apples.

We'll assume in this article that you're going to copy the entire System Master disk.

Here's how the Apple goes about making copies. First it reads a portion of the disk you're going to duplicate (often called the *original* disk), storing the information it reads in an area of the computer called *random access memory* (RAM); then it writes it on your duplicate disk (often called the *destination* disk).

The fact that the data gets stored internally between the read and write phases of the copy procedure enables you to make copies using a single disk drive. If you have only one drive, the computer will know that, and it will prompt you at the appropriate times to remove one disk and put in the other. If you have a two-drive system, it will take your Apple much less time to make the copy; you can just put the original in one drive and the destination disk in the other and watch the program go. But the copy will turn out exactly the same whether you have one drive or two.

Under the Apple Skin. Before you can start copying disks, you need to know exactly how your system is hooked up. To find out, we'll digress for a moment and take a look at the inside of your computer. If your machine is on now, you can leave it on while we do this; just be sure not to loosen anything inside while the power is on.

Reach around to the back of your Apple and apply a little pressure to the underside of the lid at the two corners. When the top pops loose, slide it straight back and out.

The largest object inside your computer, that brass or silver-colored oblong object on the left side of the Apple, is the power supply. Reach in and touch it.

It's a small formality, perhaps, but it's not a bad idea to lay a hand on your power supply whenever you have reason to go inside the computer. First of all, it's nice to know that it won't shock you; second, and more important, when you touch the power supply you harmlessly discharge any static electricity you may be carrying. If you should happen to discharge a big shock on one of the electronic components, you could damage your system.

Running from left to right across the back of your computer, there's a row of eight slots with little gold-plated connection points. These are the peripheral slots of your Apple; they allow you to expand your computer by connecting it to such devices as disk drives, printers, and modems.

If you look behind the slots, you'll see that they are numbered, starting on the left, from zero to seven. This brings up a little quaintness about computerspeak. Computer people have a habit of numbering everything from zero up, instead of starting at the number one, the way normal people do. If you've played *Little Brick Out,* you probably observed that the Apple's game paddles are called paddle zero and paddle one, not paddle one and paddle two. It's just one of those things you have to get used to.

The Rainbow Connection. Take another look at your expansion slots. You probably have a circuit card plugged into slot six (be careful not to wiggle the card; just look at it). Slot six is, of course, the seventh slot on the board, since the slot all the way over on the left is zero. You also probably have either a gray or a rainbow-colored umbilicus running from the circuit board out to your disk drive. If you have two drives, you'll have two umbilici (actually, those connectors are called ribbon cables). Oddly, the points of connection for those ribbon cables are marked drive one and drive two—not zero and one.

Be that as it may, the point of all this commentary has actually been to get you to notice which slot is holding the card that's connected to your disk drive.

The card, called the disk controller card, can be in any slot except zero, but six is a sort of standard or conventional location for it. A controller card can be connected to either one or two disk drives. If you have more than two disk drives, you'll need a second controller card, in which case the convention is to put the two cards in adjacent slots, typically six and five.

Now that you know where everything is attached, you're ready to put the lid back on and do some disk copying. Pop the System Master into your disk drive and run the appropriate program—*Copy* or *CopyA*.

The first thing that will happen is that the copy program will interrogate you a bit. It wants to know where it's going to be reading your original disk and where it will have to go to write to your destination disk.

Winning by Default. For each question, the program will provide a default answer. A default is an answer the computer assumes to be correct unless you tell it otherwise. For example, the first question asks which expansion slot holds the controller card attached to the disk drive containing your original disk. The program presumes that the card is in slot six; you can either accept the default answer by hitting return, or amend it by typing some other number.

The default values for these two copy programs will be correct if yours is a two-drive system with the controller card in slot six, and you put your original disk in drive one and your destination disk in drive two. If it happens that you have only one drive, with the card in slot six, accept the default answers to the first three questions and type "1" in response to the fourth. The computer will then know that it will be finding both the original and the destination disks in drive one and that it will have to tell you when to switch disks.

After you answer the four questions, the computer will prompt you to insert your original disk in drive one. Since the original is also the disk bearing the copy program—i.e., the System Master—you don't have to do anything at this point. Just hit return again and you're on your way.

The copy program will keep you informed about what it's doing. It will tell you, for example, whether it's reading or writing. This is important if you have a one-drive system because you could conceivably screw things up a bit by putting in the wrong disk at the wrong time. Watching the display on your screen will make that mixup less likely to occur.

Guard Your Writes. If you really want to play safe, put a write-protect tab over the disk that you're copying. That way the system can't possibly write to it, even if you have it in the drive at the wrong time. Of course, when you're backing up the System Master, you don't have to worry because the disk is already write-protected.

You'll notice that before the copy program writes anything on the destination disk, it goes through a procedure called formatting. What it's doing is structuring the disk so that the data will be laid down in an orderly fashion, enabling the computer to find it again when it's time to read the disk. The formatting process also erases anything that may have been on the disk to begin with.

When you've finished making your backup, the best thing to do is store your original in some cool, dustless place and just use the copy from now on. Use a soft felt-tip pen to label the backup, since a hard ballpoint pen could damage the disk.

If your system arrived with a disk marked "Basics," it would be a good idea to back that disk up too. We'll talk about the Basics disk in a future column.

Now that you've finished making backups, you deserve a reward for your labors, so help yourself to *Animals* or *Phone List*. Since your System Master is now copied onto a write-enabled disk, you can proceed to educate your Apple about the Kingdom Animalia or stock it full of names and phone numbers. Later on, when you've put a lot of numbers or animals on the disk, you should probably copy *Animals* or *Phone List* onto a separate disk, just to give them a little breathing room. The System Master disk is pretty full to start with; in expanding those two files, you may eventually get a DISK FULL message from your Apple.

No Copy from Scrambled Software. One more word about copying disks. Not everything can be copied with *Copy* or *CopyA*. As you may or may not know, software piracy is a big problem and a considerable threat to the microcomputer industry. To prevent indiscriminate and unauthorized duplication, quite a few commercial software publishers scramble the data on their disks in such a way that the copy programs you've just used will not be effective. Such disks are said to be copy-protected. If you try to back them up with *Copy* or *CopyA*, you'll just get an error message; no harm will come in most cases, but no copy will either.

That's frustrating, of course, if you have a legitimate need for a backup. Fortunately most of the companies that produce high-priced, copy-protected software other than games provide you with one free backup.

More about disks next time, and about that mysterious three-letter word, DOS.

MARKETALK

Impressions

from page 49

□ **Sabotage.** By Mark Allen, On-Line Systems (Coarsegold, CA). If you're tired of computer duck hunting and want to shoot down some people for a change, here's your opportunity. You man a gun emplacement with paddles or keyboard and try to knock out helicopters, bombers, bombs, and paratroopers. Mostly you've got to pick off those paratroopers. If as many as four of them can gather on the ground to either side of your cannon, they'll piggyback their way onto your battle station and zap you. Once a paratrooper gets on the ground, you ack-ack won't reach him, but you can arrange for him to be buried by falling helicopter debris or human body parts. If the saboteurs don't get you, look out for the bombers. They only come around once in a while, but they're trouble when to do. One bomb on target and you're gone. You score points according to the type of target you hit. Bombers count the most, paratroopers the least. You lose points for errant shooting, so don't fire wildly. You get to choose between normal shells and missiles that you can steer with the paddles after they've left the barrel. Colorful hi-res and sound effects. 32K, $24.95.

□ **Galactic Attack.** By Robert J. Woodhead, Siro-tech Software (Ogdensburg, NY). *Galactic Attack* is one of those rare games that combines long-range strategy and immediate arcade-type action. You are the commander of the U.S.S. *Blaise Pascal* in this hi-res Milky Way Western, charged with liberating the solar system from the savage Kzintis. To start with, the bad guys control all the planets except earth and Luna. There's even a band of those devils on the asteroid Ceres. Your job is to orbit friendly territory, beam some troops aboard, and then get those troops onto enemy-held planets. En route, you'll need either to evade or to destroy the sharp-shooting Kzinti marauders. They know your position, course, and speed, so they fire their torpedoes and phasers with deadly accuracy. And there are a lot of them and only one of you. On the other hand, you have the same weapons plus shields that the Kzintis lack. You've also got a human brain, whereas the bad guys only have a 6502 and the Apple Pascal runtime system. So the odds may be with you, but not until you've developed both strategy and tactical keyboard dexterity. 48K, $29.95.

□ **Gamma Goblins.** By Tony and Benny Gno, Sirius Software (Sacramento, CA). Here's an invader game where the shots coming at you are from little hypodermic needles with faces. You get twenty pints if you hit them. You have to wipe out the whole gang of needles, without letting any of them slip by you, before you get to face the forty-pint crew of marauding viroids. Successful lysing of the viroids, corpuscles, and lymphoids gives you a shot at the nurses' station, where a direct hit in the lower orifice yields four hundred pints and an autonomic transfusion. Odds are you won't get that far. Seven hits from this weird battalion put you in the morgue. And even if the goblins don't get you, you only have so much time to get that transfusion before your own blood supply dwindles to nothing. On the other hand, if you do make it to the second round you may encounter a couple of roving cardiacs with homing devices, who will try to zero in on you before you blast them. A great old game freshly twisted. 48K, $29.95.

CONTEST WINNERS

from page 2

camp with a strong sense of the role of computers today and tomorrow.

The California Pacific Campership went to John C. Brandstetter of South San Gabriel, California, a fourteen-year-old with a clear picture of his goals and a penchant for the pun.

First runnersup were Tim Kehoe (Plymouth, MI), twelve; Greg De Cicco (Hinsdale, IL), thirteen; Robert Allen Mason (Moorpark, CA), twelve; Sarah Robinson (Modesto, CA), ten; Ivan Drucker (Los Angeles, CA), eleven; Patrick J. Moran (Snyder, NY), twelve; Brad Handler (Denver, CO), thirteen; Mark Paul Weatherwax (Richland, MO), twelve; Geoffrey Raynor (New York, NY), thirteen; Debbie Heuerman (Fountain Valley, CA), ten; and Sharinda Hummel (Bellingham, WA), eleven.

The Oracle. The Rose Bowl and Super Bowl, the Academy Awards, and now the Indy 500 have transpired for our *Softalk* Oracles. With the scoring for guessing closest to the average speed of the winner of the Indy 500, we have a whole new order in the queue for a disk drive.

With his prediction of 141 miles per hour, Douglas Stewart (Cape Elizabeth, ME) came closest to the actual average speed of 139 for the winner of the Indianapolis 500. His score on this Oracle leg is −2. Runnersup are Craig Morris (Livingston, NJ) and Charles Lewis (Richmond, VA) who guessed 148 for −9 points.

Overall leaders so far, with their cumulative scores, are Tom O'Brien (Portland, OR), −2; Douglas Stewart and Jim Ganz (West Hartford, CT), −5; Tom O'Brien (again!) and Paul Shanberg (Moraga, CA), −8; and Daniel Tobias (Poughkeepsie, NY), −9.

Stewart opted for the cassette version of *Sargon II* as his prize.

Secret Orders. Leaving no word unturned, Tim Powers of Carrollton, Georgia, discovered eighty legitimate Applesoft commands in the March 1981 *Softalk.* There were only twelve written commands Powers didn't find. Many contestants included Applesoft reserved words that aren't specifically commands in their entries, and Powers was no exception. If these were to be counted, Powers's score would be one hundred. Either way, he won by a solid margin.

Powers will receive $100 worth of goods, including *Computer Ambush,* his primary choice, from his dealer, Compushop, compliments of *Softalk.*

Rainy Day Chain. There must have been a great deal of precipitation in Milwaukee during April and early May. S. Potts and M. Mayerhoff built us a mental roller coaster of curious associations and delightful twists and turns more than twelve hundred links long—for a total of 6,020 points before bonuses for particularly clever associations were added. Among the outstanding links in the Potts/Mayerhoff chain were: Edsel Ford/Meadowlark Lemon; Nero Wolfe/Itzhak Perlman; Arabian Nights/Sinbad/Marquis de Sade; Eydie Gorme/James Beard/Smith Brothers; Henry Ford/Model T/Lipton's; Nancy Drew/Leonardo da Vinci; That Old Black Magic/Presto, Chango/Renee Richards/Arthur Ashe. For pure puns they outdid themselves in "King Midas/It Might as Well Be Spring" and "Dolly Parton/Pardon me, boys, is that the Chattanooga Choo Choo?" And, for esoteric knowledge, we like "Halley's Comet/Mark Twain." Twain was born on the night that comet made its every-seventy-five-year appearance.

Potts and Mayerhoff will receive $100 worth of On-Line Systems games from Team Electronics of Southgate, Milwaukee, compliments of *Softalk.*

Softalk Presents The Bestsellers

In the early months of the *Softalk* Top Thirty poll, Bill Budge had so many entries on the list that it almost seemed more appropriate to name it the honorary list of Bill Budge hits. It was noted at the time that none of those programs were particularly new entries—all had been seasoned for months in the marketplace before the first *Softalk* poll, making their continued presence all the more remarkable. But another effect of the programs having aged was that none was strong enough to knock *VisiCalc* off for the top spot.

All those other Budge efforts are gone from the Top Thirty while *VisiCalc* remains, although *VisiCalc* has now been six months in the second spot. *VisiCalc* is beyond question the single most dominant program in Appledom, but lately it's taken back seat to *Alien Rain* for three months, *Space Eggs* for two months, and now *Raster Blaster,* the pinball machine simulation both authored and published by Bill Budge.

It was the first full month in distribution for *Raster Blaster* and it distanced the field. Only *VisiCalc* was even close. *Space Eggs* dropped to a far third, narrowly edging out *DB Master.*

However, the author of *Space Eggs,* Nasir, was making his presence felt elsewhere. *Pulsar II* and *Autobahn,* the two of his latest efforts available in May, both made the Top Thirty for the first time. Neither appears to have the strength to challenge *Raster Blaster* and *VisiCalc,* but Sirius Software, publisher of Nasir's programs, has high hopes for *Gorgon,* just released.

With three games in the Top Thirty and a hot new one on the way, Nasir remains the king of arcademia, that corner of the Apple world where programmers import arcade game ideas to 6502 sense. Nasir's most serious competition comes from Tony Suzuki, who has *Alien Rain* and *Alien Typhoon* in the top twenty, and *Jun Wada,* whose *Snoggle* is number five.

On-Line Systems also found May a profitable month. All three hi-res adventures from Ken and Roberta Williams made the Top Thirty as did *Sabotage* and *Missile Defense,* two arcade games authored outside the company but published by On-Line.

Previously unranked programs that hit the charts this month were lead by *Pool 1.5* from Innovative Design Software. It was eighth in its first month of wide distribution.

Sabotage, Pulsar II, Autobahn, and *Space Raiders,* from United Software of America, were also new. *Space Raiders,* also an arcade game, gives author Paul Lutus the unusual distinction of being the first author to be represented on *Softalk*'s various bestselling lists for business, home/hobby, and enter-

Business 10

This Month	Last Month	
1.	1.	*VisiCalc,* Software Arts/Dan Bricklin and Robert Frankston, Personal Software
2.	2.	*DB Master,* Alpine Software/Stanley Crane and Jerry Macon; and Barney Stone, Stoneware
3.	5.	*Supertext II,* Ed Zaron, MUSE
4.	6.	*BPI General Ledger,* John Moss and Ken Debower, Apple Computer
5.	—	*VisiPlot,* Micro Finance Systems/Mitchell Kapor, Personal Software
6.	—	*Easy Writer,* John Draper, Information Unlimited
7.	7.	*Personal Filing System,* John Page, Software Publishing Corporation
8.	4.	*Apple Writer,* Apple Computer
9.	3.	*Apple Plot,* Apple Computer
10.	10.	*Data Factory,* Bill Passauer, Micro Lab
	—	*CCA Data Management System,* Creative Computer Applications/Ben Herman, Personal Software

tainment software. Lutus previously authored *Apple Writer* for Apple Computer and *Apple World* for USA. *Apple Writer* is consistently found on the Business Ten and *Apple World*, a graphics utility package, was found in the Home/Hobby Ten.

Home/Hobby 10

This Month	Last Month	
1.	1.	*DOS 3.3,* Apple Computer
2.	3.	*DOS Tool Kit,* Apple Computer
3.	4.	*Data Capture 4.0,* David Hughes and George McClelland, Southeastern Software
4.	2.	*Typing Tutor,* Image Producers, Microsoft
5.	—	*Super Disk Copy,* Charles Hartley, Sensible Software
6.	8.	*Bill Budge's 3-D Graphics Package,* Bill Budge, California Pacific
7.	—	*Program Line Editor,* Neil Konzen, Synergistic Software
8.	6.	*LISA Assembler,* Randy Hyde, Programma
9.	—	*Multi-Disk Catalog,* Roger Tuttleman, Sensible Software
10.	9.	*DOS Plus,* Mike McLaren, Sensible Software

Apple-franchised retail stores representing approximately 8.5 percent of all sales of Apples and Apple-related products volunteered to participate in the poll.

Respondents were contacted early in June to ascertain their sales leaders for the month of May.

The only criterion for inclusion on the list was number of sales made—such other criteria as quality of product, profitability to the computer retailer, and personal preference of the individual respondents were not considered.

Respondents in June represented every geographical area of the continental United States as well as Hawaii.

Results of the responses were tabulated using a formula that resulted in the index number to the left of the program name in the Top Thirty listing. The index number is an arbitrary measure of relative strength of the programs listed. Index numbers are correlative only for the month in which they are printed; readers cannot assume that an index rating of 50 in one month represents equivalent sales to an index number of 50 in another month.

Probability of statistical error is plus-or-minus 5.1 percent, which translates roughly into the theoretical possibility of a change of four points, plus or minus, in any index number.

Programs reentering the Top Thirty were *Planetoids* from Adventure International and *Missile Defense.* Also new to the Top Thirty was *Super Disk Copy,* a venerable marketplace competitor from Sensible Software that had previously made the Home/Hobby Ten.

The Home/Hobby Ten was marked by a predominance of programmers' aids. Only *DOS 3.3, Data Capture 4.0,* and *Typing Tutor* do not fall into that category. Sensible Software had a banner month, with three of its programs making that list.

The Business Ten also underwent some changes, with three different entries on the list. Old timers returning to the list were *Easy Writer* from Information Unlimited and *CCA Data Management System* from Personal Software.

New to the list and the marketplace was *VisiPlot* from Personal Software. *VisiPlot* is sold both as a stand-alone package and in concert with *VisiTrend.* The ranking here is for *VisiPlot* as a stand-alone package; if its sales with *VisiTrend* had been counted, it would have been third in the Business Ten.

Business in general in May held even with the April upturn without showing significant advancement. About as many retailers reported a slowdown in business as reported an upturn.

This market strength is being reflected in broader-based support for a wider range of product than had previously been the case. There are a far larger number of software products just below the last entry in each list than ever before. ■

The Top Thirty

This Month	Last Month	Index	
1.	1.	96.45	*Raster Blaster,* Bill Budge, BudgeCo
2.	2.	84.21	*VisiCalc,* Software Arts/Dan Bricklin and Robert Frankston, Personal Software
3.	1.	55.24	*Space Eggs,* Nasir, Sirius Software
4.	5.	51.06	*DB Master,* Alpine Software/Stanley Crane and Jerry Macon; and Barney Stone, Stoneware
5.	4.	47.48	*Snoggle,* Jun Wada, Broderbund Software
6.	8.	45.69	*Hi-Res Adventure #2: The Wizard and the Princess,* Roberta and Ken Williams, On-Line Systems
	10.	45.69	*DOS 3.3,* Apple Computer
8.	—	41.21	*Pool 1.5,* Don Hoffman, Howard de St. Germain, and Dave Morock, Innovative Design Software
9.	6.	39.71	*Alien Rain,* Tony Suzuki, Broderbund Software
10.	11.	37.92	*Zork,* Mark S. Balnk, Timothy Anderson, Bruce Daniels, P. D. Leblins, Scott Cutler, and Joel Berez/Infocom, Personal Software
11.	—	37.03	*Sabotage,* Mark Allen, On-Line Systems
12.	7.	35.83	*Flight Simulator,* Bruce Artwick, SubLogic
13.	17.	33.14	*Olympic Decathlon,* Tim Smith, Microsoft
14.	—	26.58	*Pulsar II,* Nasir, Sirius Software
15.	9.	25.98	*Warp Factor,* Paul Murray, Strategic Simulations
16.	—	25.38	*Space Raiders,* Paul Lutus, United Software of America
17.	—	23.88	*Autobahn,* Nasir, Sirius Software
18.	24.	23.59	*Alien Typhoon,* Tony Suzuki, Broderbund Software
19.	15.	22.99	*DOS Tool Kit,* Apple Computer
20.	—	22.40	*Missile Defense,* Dave Clark, On-Line Systems
21.	12.	21.50	*Sargon II,* Dan and Kathe Spracklen, Hayden
22.	13.	19.11	*ABM,* Silas Warner, MUSE
23.	26.	17.62	*Hi-Res Adventure #1: Mystery House,* Ken and Roberta Williams, On-Line Systems
24.	16.	17.32	*Hi-Res Adventure #0: Mission: Asteroid,* Ken and Roberta Williams, On-Line Systems
25.	—	15.83	*Planetoids,* Marc Goodman, Adventure International
26.	19.	15.53	*Data Capture 4.0,* David Hughes and George McClelland, Southeastern Software
27.	14.	14.63	*Typing Tutor,* Image Producers, Microsoft
28.	21.	14.03	*Supertext II,* Ed Zaron, MUSE
29.	26.	13.14	*BPI General Ledger,* John Moss and Ken Debower, Apple Computer
30.	—	12.84	*Super Disk Copy,* Charles Hartley, Sensible Software
	21.	12.84	*The Prisoner,* David Mullich, Edu-Ware Services