Recreational COMPUTING

FOR THE IMAGINATIVE SMALL COMPUTER USER!
VOL. 9 NO. 5 ISSUE 50 MAR-APR 1981 $2.50 In Canada $3.00

SPACE EXPLORATION

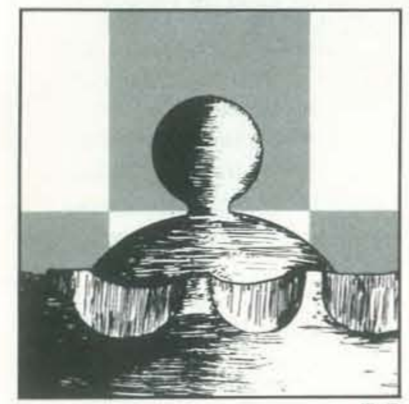
Frontiers for You and Your Micro

VOYAGE TO ANTARES

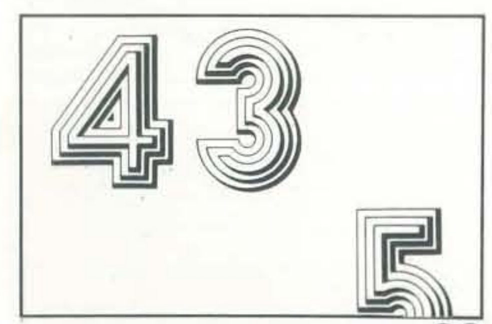
A SPACESHIP SIMULATOR

STAR TREK
A Dialogue Approach
Part II

8



16



29

Cover: Photos courtesy of NASA.
Montage by Dave Peebles

RECREATIONAL COMPUTING (ISSN #0164-5846) is published bimonthly by People's Computer Company, 1263 El Camino Real, Box E, Menlo Park, CA 94025. People's Computer Company is a non-profit, educational corporation. Donations are tax deductible. Second class postage paid at Menlo Park, California, and additional entry points. Address correction requested. Postmaster: send form 3579 to Box E, Menlo Park, CA 94025.

Recreational COMPUTING

March/April 1981, Volume 9, Number 5, Issue 50

SPACE EXPLORATION

RECREATION

DEPARTMENTS

Recreational COMPUTING

March/April 1981
Volume 9, Number 5
Issue 50

Publisher
Ann Merchberger
Editor
Marlin Ouverson
Editorial Assistants
Julie Anton
Pat Cleland
Michael Madaj
Contributing Editors
Dave Caulkins
Dave Cortesi
Ramon Zamora
Art Director
Clifford West
Artists
Barbara Ruzgerian
Greg Gordon
Typesetter
Renny Wiggins
Proofreader
Nancy Heubach
Marketing Director
Gavin W. Cullen
Advertising Director
Michael Madaj
Circulation Manager
Peter Clark
Circulation Assistant
Leah Dansby

Subscription rates: $12 per year within the United States; $20 for first class to Canada and Mexico; $26 for airmail to other countries. Payment must be in U.S. dollars, drawn on a U.S. bank.

Contributing Subscribers: $25/year ($13 tax deductible) – Algorithmics, Inc., DeWitt S. Brown, Gerald Bowman, Robert Connors, David R. Dick, Mark Elgin, Joi Ellis, John B. Fried, Scott B. Guthery, Alan Hamilton, Brian Herring, T. Alton Howard, William G. Hutchison, Jr., W. A. Kelley, Land of Light, William M. Richman II, Phillip A. Smith, Neil Sullivan, Joseph A. Weisbecker, Brett Wilson. **Retaining Subscriber:** $50/year ($38 tax deductible) – Dave Caulkins. **Sustaining Subscribers:** $100+/year ($88+ tax deductible) – Byte Publications; Paul, Lori & Tom Calhoun; Louis R. Patzke. **Lifetime Subscriber:** $900+ ($700+ tax deductible) – Bill Godbout Electronics. **Corporate Subscriber:** $500/year ($440 tax deductible.

Advertising: Advertising space is available in this publication. Please direct inquires to the Advertising Director, *Recreational Computing*, Box E, Menlo Park, CA 94025.

Foreign Distributors of ***Recreational Computing*****: UK & Europe:** L P Enterprises, 8/11 Cambridge House, Cambridge Road, Barking, Essex, 1G11 8NT, Great Britain. Hofacker-Verlag, Tegernseer Strasse 18, D-8150 Holzkirchen, West Germany, Computerland/Computer Store AB, Box 7134, Kungsgatan 19, S-10387 Stockholm, Sweden. **Canada:** RS-232, 186 Queen Street W., Toronto, Ontario M5V 1Z1, Canada. **Asia & Australia:** Electronic Concepts Pty Ltd., 55 Clarence Street, Sydney, NSW 2000, Australia. Computer Store, POB 31-261, 22B Milford Road, Milford, Auckland 9, New Zealand. ASCII Publishing, 305 Hi Torio, 5-6-7 Minami Aoyama, Minato-ku, Tokyo 107, Japan.

Editorial

Tales of Power

The microprocessing field is changing. An obvious remark, on the surface. Also an understatement? Undoubtedly.

Small computers and related technologies have continued their explosive, years-long emergence from translucent, silicon cocoons. Lower costs and wider accessibility are now bywords. New and better chips, smaller and faster processors, and more efficient languages are always news, *de rigeur* among annual product unveilings. This is a domain where innovation has become an institution.

The last decade has seen homebrew hobbyists knocking the walls out of their garages one day to find themselves barons of multi-million dollar corporations of international influence. Micros survived birth into the grey-flannel-suit world and so, for better or worse, did their creators. Watching these most recent survivors of the initiatory rites into the world of high finance, corporate high priests have now developed a taste for silicon.

It is not news that a number of magazines for computer consumers and hobbyists have been bought out by large publishing companies. It will, however, be news to many that Tandy Corporation has signed a contract (non-exclusive!) naming Random House, itself a subsidiary of RCA, an authorized distributor of TRS-80 products for educational use; they will be developing related software. Another item is that England's BBC will blazon its initials on a small computer; users will attach the inexpensive unit to their televisions in order to fully participate in a series of educational programs. And in France, the national telephone company has found it will be less costly to install remote terminals in private homes than to print periodic directories; users will have nearly immediate access to updated information.

The upshot of all this is that micros make good business sense on a very large scale. Even in the short term, we will be seeing many more big business fingers in the pie. And, given the power of information processing, whoever holds the key to micros also holds the key to the considerable power which will result from their use in broad applications.

A balance must be struck, and it will indeed be a balance of power. The open flow of communication must not be restricted, regardless if it uses telephone lines or airwaves for its medium of transmission. Operations must not be so complex as to stifle wide participation and understanding. Overworked though the phrase may be, "computer literacy" must find a place in the home of each individual who wishes to maintain the maximum control of his own life.

The implications of today's events in microprocessing will, in any case, be far more profound than we can foresee.

Marlin Ouverson
Editor

Letters

A big, editorial "Thank You" goes to all those readers who responded to a recent *RC* survey. It has helped us to take a good, objective look at ourselves and to know how we can serve you better.

A free one-year subscription is being sent to the five randomly selected respondents listed below:

Gary Bitter
Marc Edgar
R. L. Rathbun
William J. Shaw, Jr.
John G. Singer

Dear Editor:

Although I do upon occasion write to publications regarding their material, it is not my style to criticize in such correspondence. As a writer myself, I am highly sensitive to the effect of readers' reactions on one who has expended great energy and personal creativity to prepare a piece for publication.

However, in the case of the "House of the Future," (*RC* #48) I MUST cry out in protest, not against the work of the author, for the article was beautifully written and illustrated, but against the quality of the structure itself. A house must be judged by other criteria in addition to technology, and even some of the technology of this one falls flat. Inspection uncovers sham pertaining to some of the developers' claims, a cloudy advertising technique forced upon the author himself, as evidenced by the vague allusions to what the house's computer system WILL do, as opposed to what it does in fact perform today.

The microprocessor is the biggest fib of all. Prompted by my companions, a workshop comprised primarily of architecture students and architects, as the only computer professional among us, I naturally sat down at the terminal shown on page 10 of your November-December issue. The menu printed on the screen invited the home user to schedule or instruct various appliances, as well as utilize other routine people aids, such as enter reminders for future days, shopping lists, etc. Nothing too futuristic there, for sure.

However, no sooner had I sat down, pressed the number of my selection, and watched the very same menu reprint itself on the screen when two huge guards converged upon me, strongly urging that I leave the room. As I weigh 90 pounds, I was not about to argue. In another room, I again addressed a terminal. The guard in there exhibited a bit less 1980ish paranoia. He asked me to stop, explaining that nothing in the house was actually hooked up to the computer except the refrigerator on/off cycle, activated to coincide with days the guards themselves were present and might want to refrigerate their lunch bags.

Claims for the environment, as outlined in the article by Motorola, are only half-true. The internal termperature of the house, is, in fact quite pleasant, a coup deserving of applause, considering the stifling August heat which surrounded the future abode on the day that I visited. However, internal lighting, despite the ample Arizona sun, proved too dim for safe, continuous, concentrated eye use, such as one would need for reading.

The welcoming voice at the door bore the quality of an audio tape, not computer-generated sound, and the key pad entry, too, was far less sophisticated than would require digital computer, or than is found in any office requiring entrance security. It seemed to act more as a combination lock on a suitcase. Even if perused by the microcomputer, the security system bore more familiarity as a typical home computer application than a flavor of fantasy for the future.

When my companions and I inquired as to the selling price of the house, we were quoted some astronomical price for that area of the city, and told that $50,000 of that was for the software. Obviously, that $50,000 would be utilized by Motorola to DEVELOP software AFTER the house is purchased, since no widely distributed microcomputer-bound software system to date had carried a price tag so disproportionate to hardware costs. The next houses would evidently benefit from the investment of the pioneer owner. This pricing technique unfortunately pervades the software vendor world.

Systems purported to satisfy the custom needs of the first purchaser are then widely marketed as packages to large user bases which dilute the costs and lower the price. Of course, an initial outlay must be provided for system development, but the developer would benefit the consumer greatly with an open admission of this necessity.

In the case of Ahwatukee, the middle-class neighborhood of its surroundings will not attract the kind of resident able to invest such funds. Since the house is both buried deeply in the ground, and exhibits so-called "middle-class" taste, one assumes it will not be moved to a neighborhood where it would attract appropriate support.

I've had my say. May I suggest that the visitor to the "House of the Future" also take the time to travel Northward, first to Taliesen, in Paradise Valley, to see how Frank Lloyd Wright meant the principles he espoused to be translated, and then to Arcosanti, at Cortes Junction, where Paolo Soleri, former, though disenchanted student of Wright's, has applied many of the same principles of environmental control via cast concrete construction, solar principles, recirculated water, air, etc. and the greenhouse effect for temperature control that one finds in Ahwatukee. You will find a city free of technology, poured by hand by volunteers, where acoustics and natural lighting combine with perfection as a backdrop for the performing arts, in addition to a growing residential community free of automobiles and pollution. These two divergent structures, emanating from the origins of one man, each employing the most avant-garde architectural principles of its day, spark the viewer to understand that variety and choice are due the dweller. Neither Wright nor Soleri should dictate environmental design for everyone. A mix of structures, ranging through time, incorporating the wishes of the dwellers themselves, enrich our heritage, enhance, or at least coexist in partnership with our natural landscapes and restore the greatness of our country. Technology is a tool, not an objective, to be applied where helpful in improving our standard of living, as we, the living choose.

As to the non-existant Ahwatukee computer system, enough expertise exists among the readership of *Recreational Computing* to develop bits of the required software, at home as usual, distributing it not only to this one "House of the Future," but for general use at the affordable price permitted by such wide disbursal. Nothing so esoteric exists in the plans for Ahwatukee that such a system would lack wide appeal. And after all, what software affects more acutely your recreation than that which controls your home environment?

Abby Gelles,
Author, *Robotics Curriculum*
Arcosanti Workshopper, Summer 1980
185 West Houston St.
New York, NY 10014

Dear Sirs,

The July-August 1980 issue was the first time I had an opportunity to read your magazine and I enjoyed it. I am a novice TRS-80 programmer and also a seventh grade teacher who wants to involve his class in programming and in the logic of adventuring. I liked the article by Dr. Furman Smith on CFS, but it was a little too complex for my use.

What programming skills are needed to develop a short CFS? (Four or five rooms, some treasure, a monster that can be killed with a weapon only obtained by meeting other requirements.)

The article mentioned an "enchanted house" program that was general and easy to modify. Is it really easy to understand? If so, how can I get a listing of it?

Are there any other articles or books for the *beginning* basic adventurer?

Are there any workbook-texts that would be useful for my seventh grade class? I would like to introduce programming to some of my students who lack a math background and I'm looking for non-threatening programming problems.

Thank you for your help.

James Crawford
2802 Agua Vista
San Jose, CA 95132

Sir,

I was pleased to see your publication of my article "Dozo in Pascal" in your November/December issue of *Recreational Computing*. Unfortunately, in reviewing the article, I noticed several editing errors which will prevent the program from compiling properly. I have enclosed an edited copy of the program as listed in your magazine and another listing of the first part of the program.

The first and most obvious error is that the program starts three times on the first page. The program begins with the line "(*$5*)" followed by the line "PROGRAM DOZO;". The program appears to be reproduced properly beginning with the third incidence of these lines. The other error in the listing occurs at the end of the procedure "SETLOCS." There must be a second "END;" line immediately following the existing "END;" line in the SETLOCS procedure and immediately before the line "PROCEDURE BLOCK (. . . . . ." Both of these errors are fatal to the program. While the first error may be obvious to the experienced Pascal programmer, the second error would leave them wondering what other lines may have been omitted.

I hope that you will be able to publish a correction of these errors so that your other readers will be able to enjoy using the program as much as I enjoyed writing it.

Thank you.

Stephen R. Berggren
2347 Duncan Dr. #4
Fairborn, OH 45324

Following is the correct version of the first section of Mr. Berggren's program. Our apologies! – Ed.

```
(*$5+*)

PROGRAM DOZO;

USES TURTLEGRAPHICS, APPLESTUFF;

CONST  NUMTRI = 126;(* # OF WINNING TRIANGLES *)
       NUMPOS = 28;(* # OF BOARD POSITIONS *)

TYPE   POSITION   = 1. .NUMPOS;
       TRIAN      = 1. .NUMTRI;
       THREE      = 1. .3;
       VALCOLOR = 0. .10000;

VAR    WIN, WON, NWIN, TIED, FMOVE : BOOLEAN;(* GAME
       CONTROL VARIABLES *) TRIWIN, TRIWON : TRIANG;
       (* WIN=COMPUTER, WON=PLAYER *)
       WINCOLOR, WONCOLOR : VALCOLOR;
       WINPOS, GMOVE : POSITION;
       CH : CHAR;
       TRIANGLE : ARRAY[TRIANG, THREE] OF VALCOLOR;
       (* COLOR OF THE CORNERS OF EACH TRAINGLE *)
       CORNER : ARRAY[TRIANG, THREE] OF POSITION;
       (* BOARD POSITION OF CORNERS OF EACH TRIANGLE *)
       BOARDPOS : ARRAY[POSITION] OF VALCOLOR;
       (* COLOR OF EACH BOARD POSITION *)
       XPOS, YPOS : ARRAY [POSITION] OF INTEGER;
       (* X, Y SCREEN POSITION OF EACH BOARD POSITION *)
       DOZOFILE : FILE OF CHAR;
       (* HOLDS THE BOARD POSITIONS OF CORNERS OF EACH
       WINNING TRIANGLE *)
       I,J,MOVES : INTEGER;

PROCEDURE FILLTRI;
(* FILL CORNER[ ] FROM DOZOFILE.TEXT *)

VAR I,J : INTEGER;

BEGIN
  RESET(DOZOFILE,'DOZOFILE.TEXT');
  FOR I := 1 TO 3 DO
  FOR J := 1 TO NUMTRI DO
  READLN(DOZOFILE,CORNER[J,I]);
  CLOSE(DOZOFILE);
END;

PROCEDURE SETLOCS;
(* FILLS XPOS,YPOS WITH SCREEN POSITIONS *)

VAR A,I,ROW : INTEGER;

BEGIN
  A := 1;
  FOR ROW := 1 TO 7 DO
    FOR I  := 1 to ROW DO
      BEGIN
        YPOS[A] := (7-ROW)*23+1;
        XPOS[A] := ((I*2)+6-ROW)*15+45;
        A := A+1;
      END;

END;

PROCEDURE BLOCK(X,Y: INTEGER;DRAWCOLOR: VALCOLOR);
(* PUTS A COLORED BLOCK ON THE SCREEN *)
BEGIN
  VIEWPORT(X,X+10,Y,Y+9);
  IF DRAWCOLOR = 0 THEN FILLSCREEN(WHITE2);
  IF DRAWCOLOR = 1 THEN FILLSCREEN(BLUE);
  IF DRAWCOLOR = 10 THEN FILLSCREEN(GREEN);
  IF DRAWCOLOR = 100 THEN FILLSCREEN(ORANGE);
  IF DRAWCOLOR = 1000 THEN FILLSCREEN(VIOLET);
  IF DRAWCOLOR = 10000 THEN FILLSCREEN(BLACK2);
  VIEWPORT(0,279,0,191);
END;

PROCEDURE DRAWSIDES( TRIIN : TRIANG; COLORIN : VALCOLOR);
(* DRAWS A TRIANGLE *)

VAR INX,INY : INTEGER;

BEGIN
  INX := XPOS[CORNER[TRIIN,1]]+5;
  INY := YPOS[CORNER[TRIIN,1]]+4;
  MOVETO(INX,INY);
  IF COLORIN = 1 THEN PENCOLOR(BLUE);
  IF COLORIN = 10 THEN PENCOLOR(GREEN);
  IF COLORIN = 100 THEN PENCOLOR(ORANGE);
  IF COLORIN = 1000 THEN PENCOLOR(VIOLET);
  INX := XPOS[CORNER[TRIIN,2]]+5;
  INY := YPOS[CORNER[TRIIN,2]]+4;
  MOVETO(INX,INY);
  INX := XPOS[CORNER[TRIIN,3]]+5;
  INY := YPOS[CORNER[TRIIN,3]]+4;
  MOVETO(INX,INY);
  INX := XPOS[CORNER[TRIIN,1]]+5;
  INY := YPOS[CORNER[TRIIN,1]]+4;
  MOVETO(INX,INY);
  PENCOLOR(NONE);
END;

PROCEDURE INSTRUCT;
(* GIVES INSTRUCTIONS *)

VAR CH : CHAR;

BEGIN
  TEXTMODE;
  WRITELN;
  WRITELN;
  WRITELN('            DOZO');
  WRITELN;
  WRITELN;
  WRITELN('THE OBJECT OF THE GAME IS TO CREATE AN');
  WRITELN;
  WRITELN('EQUILATERAL (EQUAL-SIDED) TRIANGLE BY');
  WRITELN;
  WRITELN('PLACING THREE BLOCKS OF THE SAME COLOR');
  WRITELN;
  WRITELN('AT THE CORNERS. TO MAKE A MOVE, USE');
  WRITELN;
  WRITELN('PADDLE(0) TO MOVE THE FLASHING BLOCK');
  WRITELN;
  WRITELN('TO THE RIGHT PLACE. THEN PRESS THE KEY');
  WRITELN;
  WRITELN('FOR THE COLOR YOU WANT.');
  WRITELN;
  WRITELN;
  WRITELN('       (PRESS ANY KEY)');
END;
```

Dear Editor:

Would you please consider including the following correction in a future issue? Thank you.

The magic square printed on page 42 of the November-December *RC* contains a typographical error. The number in the upper left corner of the magic square should be 139 instead of 131.

Sincerely yours,
Allan Wm. Johnson Jr.

Dear Sir:

One day about three years ago I decided to stay after school to try out the school's computers. I didn't know what I was doing! Three times that day (in about half an hour) I shut down the entire computer system! After reading a lot of books, I found out how to operate the system, *perfectly!* Then the school got different computers.

They got several 8K Commodore (PET) computers. Only these didn't use BASIC language, they used New BASIC. Now I am a full-fledged computer buff. After using these computers I decided to get one of my own, an Atari 400 8K (BASIC language) computer. I need $500.00 to buy it. So far I have $200.00 saved.

As you probably know by now, when I write I get away from the main idea. Well here it is: my computer teacher showed me a book of computer programs for games. It was titled, *What To Do After You Hit Return.* I would like to know how and where I can obtain a copy of this book, or any other computer games books in the BASIC language. Also, please send me any pamphlets you have concerning computers or computer programming.

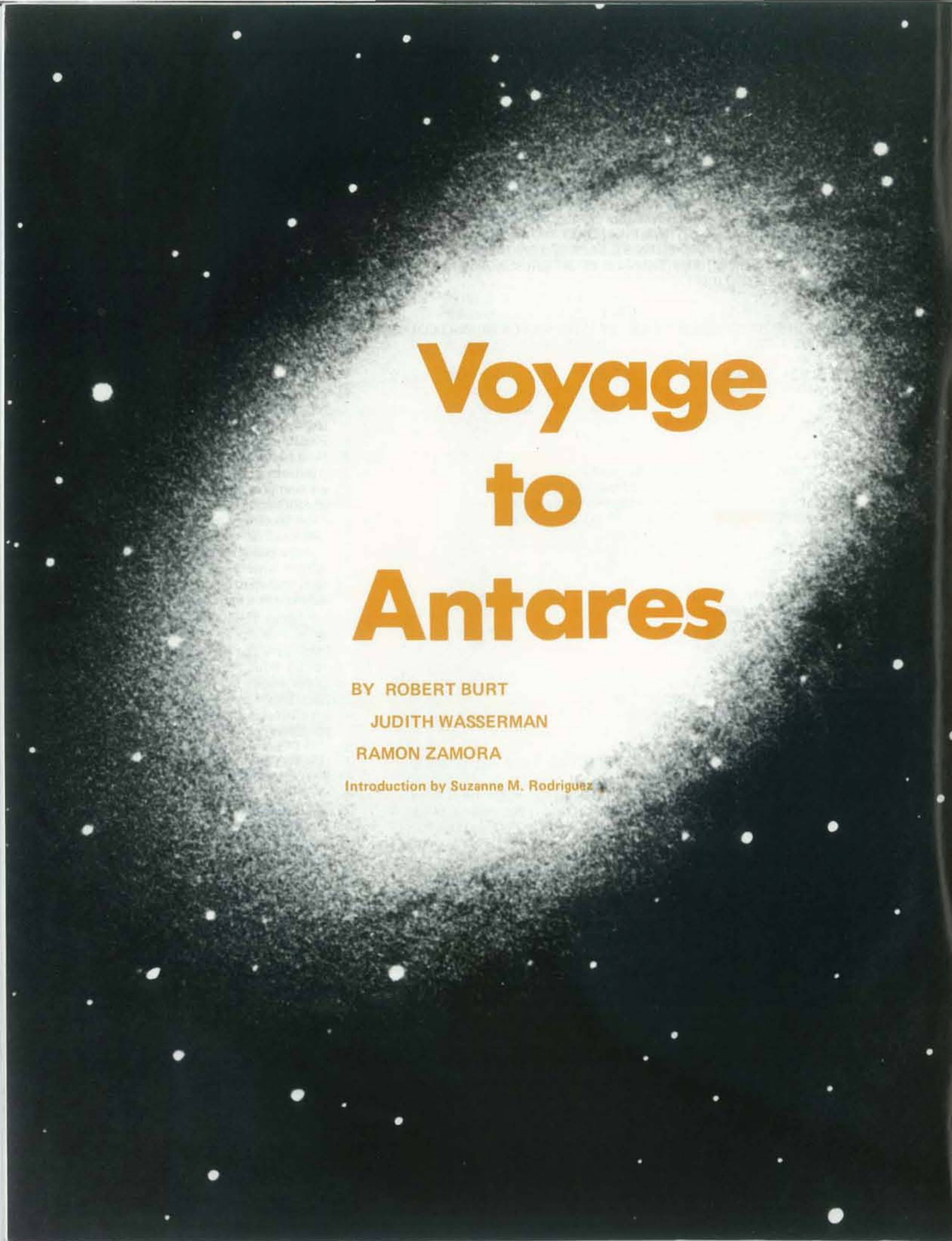
Thank you for the patience and time that you spent reading this letter,
Fred Derenthal

Dear Fred,

What To Do After You Hit Return *is currently handled by Hayden Book Company, Inc., 50 Essex Street, Rochelle Park, NJ 07662 and can be purchased for $14.95. People's Computer Company, P.O. Box E, Menlo Park, CA 94025 does have some copies of the 1979 edition available for $10.95.*

Games and recreation books for both the PET and the Atari will be available soon from Reston Publishing Company, 11480 Sunset Hills Road, Reston, VA 22090.

Other books you may want to take a look at are the following: Stimulating Simulations *by C. W. Engel, published by Hayden;* BASIC for Home Computers, Atari BASIC, *and* TRS-80 BASIC *by Dymax Authors and published by John Wiley & Son, Inc., 605 Third Avenue, New York, NY 10016. I understand that they have an Apple games book in progress. – Ed.* ■



Voyage to Antares

BY ROBERT BURT

JUDITH WASSERMAN

RAMON ZAMORA

Introduction by Suzanne M. Rodriguez

Many years ago, the British savant C. P. Snow, in his book The Two Cultures, *warned of the danger of separating art from science. Snow, a scientist turned writer, decried this tendency toward polarization. Only through the constant mingling of ideas and perspective, he felt, could a brilliant creativity survive in the world.*

It's probably true that there are those, both in science and the arts, who deny viability of expression to the opposite culture. We've all known artists who dismiss technology as boring and life-annihilating; or the technology buff who's never read a work of fiction or studied a painting. They miss a lot: at the very least a greater appreciation for the world around them and a heightened perspective. I am reminded of my undergraduate days at Stanford when the physicist Wolfgang Panovsky addressed a class of mine. He said if he didn't play his violin at least an hour a day, his creativity would dry up.

What's particularly remarkable about this article is that all three contributors combine technological expertise and artistic know-how.

The pilot, Robert Burt, has an M.S. in physics from NYU and an M. A. in music from Columbia. Bob is a Research Specialist at Lockheed's Space Systems Division. While addicted to classical music, he has recently taken an interest in computer music.

The artist, Judith Wasserman, has a B.A. in physics from Cornell University. For the last several years, she has devoted her time to artwork, especially etching and lithography. A member of Palo Alto's successful art cooperative, Gallery House, she does layout work for InfoWorld.

The critic, Ramon Zamora, has a B. A. from the Florida Institute of Technology. With sixteen years of computer experience behind him, Ramon is Project Director for ComputerTown, USA! and Vice President of Avalanche, Inc. in Palo Alto. Co-author of a series of introductory books on microcomputers, Ramon also acts as a Contributing Editor to this magazine.

One note of caution before you proceed with the article. Bob Burt developed his ByByBaybee Program on an HP 9830, but he gives enough information to get the small computer user underway. Unlike most of our articles, then, this one does not give you all the answers: it is meant to excite your creative palate, to get you up and running. Have fun with this one, and good luck!

The Critic

At times, computers are said to generate art (graphic forms, textile design patterns). At times, artists (engineering artists, design artists, software artists – yes, there is art involved) generate computers. Once in a while a quiet synthesis occurs when a person working in an area of the fine arts and information from a computer program are joined together.

A show called "Visions of Flight," in the Gallery House of Palo Alto, California, contained such a point of synthesis. Judith Wasserman in her section of the show, subtitled "Voyage to Antares," displayed a series of relief etchings derived from the output of a computer program. The program, written by Robert Burt, generated the star configurations that would be observed if one were to travel along a line from our sun to Antares.

Photo courtesy of NASA.

For the show, points along this space path at 10, 50, 200 and 500 light years were selected. Haunting views seen from a spacecraft's window fill each print. The cold blackness of space; the bright points of light that represent the stars; the faint yellows, oranges and reds of distant suns. And then, embossed in white across the surface of the prints, renderings of mythical creatures formed of star points and dark voids. *The Bearded Bull of Canopus*; *The Great Octopus of Antares* hang in space 500 light years from here. *The Horned Cats of Hadar*; *The Serpent of Spica* prowl the stars 50 light years away. In all there are fifteen creature/myths portrayed. In the words of the artist they represent "an expansion of earthbound mythology into a future time and space . . ."

It is joyful to see and experience this creative marriage of technology and spirit, of data and insight. The question arises as to who else and where else is this being done.

The Artist

I first became interested in a program like ByByBaybee when I thought I'd like to go to the center of the Galaxy on my next vacation. Then I met Bob Burt at a party on New Year's Day and discovered he was a celestial mechanic. I had found my pilot. He knew programming, he knew computers, he knew astronomy, he liked art and he was very interested.

My idea was to design a program that would print out a chart of stars as seen from any point in the Galaxy, then to pick a destination, make several stops and draw the constellations we could see along the way.

We picked Antares because it was toward the center of the Galaxy and was particularly beautiful from Earth, being red and sparkly, and interesting from close up, being a red giant and very luminous.

Bob spent the spring and summer building the ship and we decided how many stops we could make and at what distances. Since each stop meant a complete sky plot, it entailed a fair amount of computer time, so we limited the number of stops to four. By this time I had a date lined up on which to exhibit renderings of these constellations and I thought I could find at least three images in each distance, which, along with a few other related pieces, would be enough to both show and exhaust my concept.

The print-out was delivered to me: one 11" x 15" piece of graph paper with 286 x 4 diamond shapes in four colors (a different color for the perspective viewed from each of the distances selected) and assorted sizes (denoting varying intensities). Actually, some of the stars were too dim to show up at 200 and 500 light-years. The chart in itself was beautiful in four colors, one color for each distance, but in order to find constellations I first had to trace off the stars at each distance separately. Then I could look at them up, down and sideways; sometimes I found it necessary to project something not quite there.

One thing missing from this trip to Antares was the sighting of new stars, since the computer couldn't print out stars we didn't put in. Consequently, by the time we got to 500 light-years, the sky was almost empty. I compensated for this creatively by adding fields of imaginary stars. I made a visual distinction between "real" or computer-generated stars, and imaginary ones: I made the real ones larger and varied their size according to their computer-determined relative luminosities, whereas the imaginary stars were just pinpricks and swarmed in arbitrary or artist-generated patterns over the image.

In the end I had twelve constellations, three at each distance (10, 50, 200 and 500 light-years), named after the brightest star in each group. They were hand printed as relief etchings with white lines and stars on dark fields and had names like *Serpent of Spica* and *Sphinx of Arcturus*.

The Pilot

Background to the Stars. The annual voyage of the earth around the sun is miniscule in size compared to the enormous distances from our solar system to the nearest stars. Consequently, the differences in the geometric patterns of the stars as seen from opposites sides of the sun are too minute to be discernable to unaided earthling eyes. These differences were also too small for detection by the instruments, however sophisticated for their day, of the gifted astronomers of ancient Greece, the Renaissance, and the first decades of the telescope. These scientists carefully measured the angular separation of the stars and plotted these light points on a celestial globe, but found no semi-annual change in the patterns of the constellations.

At the time of Copernicus, and for many years thereafter, the absence of detectable shift in the patterns of the constellations as seen from opposite sides of the sun was the strongest scientific argument against his heliocentric hypothesis. It was difficult to believe that stellar distances were so great that semi-annual star shifts were below detectable measurement. Furthermore, if these freakish distances were correct, then incredible strengths of outpouring luminous flux would be required of these "fires" of heaven if they were to be seen from as far as earth. Thus, two unbelievable concepts, incredible star distances and gargantuan light sources (both true), accompanied the Copernican challenge to the old belief in a fixed earth at the center of the universe.

Acceptance of these enormities required much time and new generations of earthlings.

Ultimately, increasing evidence, more sophisticated planetary theory and mathematical simplicity forced the adoption of the Copernican view and carried the implications of stellar hugeness along with it.

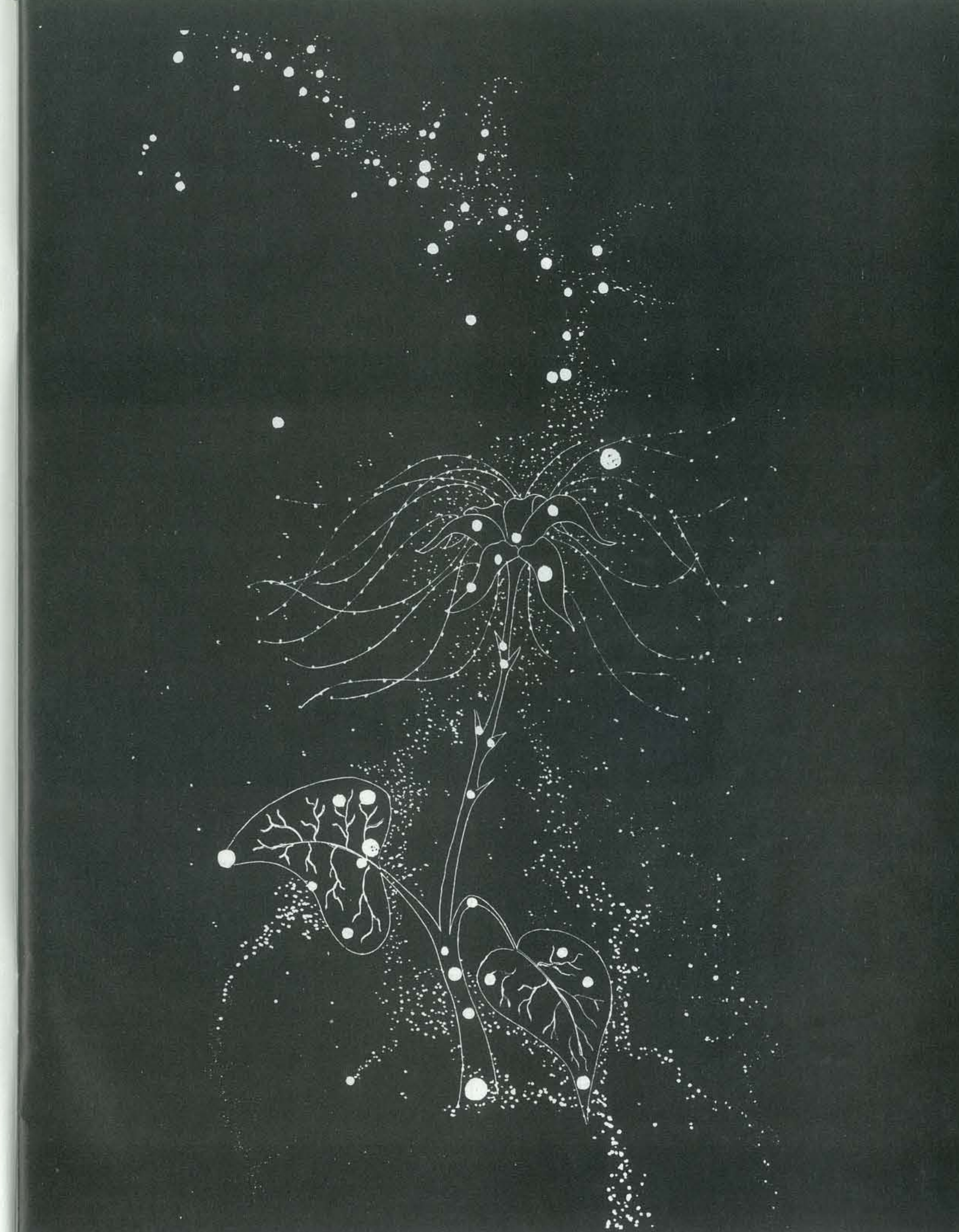
The search for semi-annual star shifts continued. These shifts would tell how far away the stars were. Toward the middle of the nineteenth century, three centuries after Copernicus, precision telescopic instrumentation was sufficiently developed. In South Africa, Thomas Henderson measured the greatest possible shift: that of the nearest star, α Centauri. The shift was 1½ seconds of arc or about 1/1200 the diameter of the moon! The next nearest star, Barnard's Star, has a shift of just over one second of arc; all other star shifts are below one second of arc!

So you see, dear reader (are you still there?) that to observe a substantial change in the patterns of the constellations, sufficient to interest an artist and to cause this artist to imagine new shapes and beings that inhabit the transformed heavens, we must leave the solar system far behind with the sun ultimately at our rear no matter the direction we travel. We must take a gargantuan voyage through our galaxy (the Milky Way), gargantuan to us, but quite small compared to the size of the galaxy. In our first excursion, we observe fewer than 300 stars in a small suburb of a galaxy that has an estimated two billion stars.

And finally, a confession to two omissions: We have included only the brightest stars of our heavenly experience on earth and not many dimmer stars that may well become prominent in a galactic voyage. And we have assumed that in our travels, perhaps over many generations, the stars will remain motionless – which is not quite true.

Overview. The computer program ByByBaybee calculates and plots the altered patterns and visual magnitudes of the stars as seen by a space traveler at a specified distance from the sun on the path to a specified star. The program, written in BASIC for an HP 9830 and associated plotter, is in three parts. The first part swallows, processes and stores star data in six memory files. The second part retrieves the data and plots a conventional star map for checkout purposes. The third part retrieves the data (one file at a time to prevent core overload) and calculates and plots the new orientations and visual magnitudes (star brightness) at the given spacecraft location.

Change in Star Patterns. The mathematics of the change in star patterns is the mathematics of three-dimensional perspective. The stars diverge from the point where the extended line of spacecraft motion pierces the heavens (pole of approach) and converge toward the opposite pole, the vanishing point of the artist. If the spacecraft moves along a straight line, the apparent



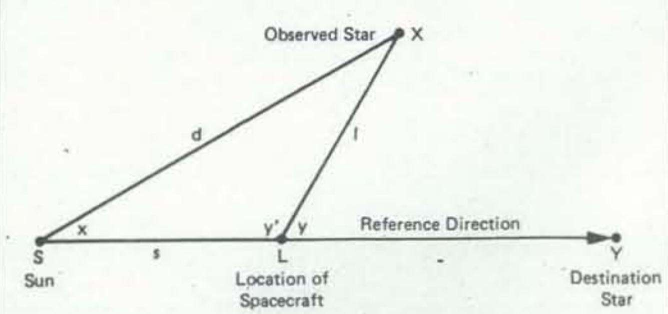
This untitled lithograph and the others of Creature/Myths in this article are by Judith Wasserman.

The Crested Eagle of Vega

positions of the stars in the celestial sphere centered at the spacecraft move along half great circles which arise at the pole of approach and terminate at the vanishing pole. The rate of travel along these half circles is slower for the further stars and varies for each star, being fastest when the spacecraft passes the point of closest approach to the star.

Two-Dimensional Analysis. The two-dimensional analysis of the altered direction of a star is illustrated in Figure 1. Points S, L, Y and X are the solar system (at these interstellar distances, the solar system is a mere point), spacecraft location, destination star and observed star, respectively. Angle x is the direction of the observed star at S with respect to the direction of the destination star (taken for convenience as reference direction), and d is the distance from S to X.

The problem is to find angle y, the direction of the observed star at various distances, s, from the solar system on the path to Y, and to find l, the distance of X from L. These are needed, together with the given absolute magnitude, to compute the visual magnitude of the star as seen from L. The solution is simple. We are given three consecutive parts of triangle XSL: distance d, angle x and distance s. Angle y′, the supplement of desired angle y is the fourth consecutive part of the triangle; it is obtained by means of the four-parts law: $y' = \tan^{-1} [d \sin x/(s - d \cos x)]$. Distance l is obtained from the given data by the law of cosines: $l = (d^2 + s^2 - 2ds \cos x)^{1/2}$.



$$y = 180 - \tan^{-1} \frac{d \sin x}{s - d \cos x}$$

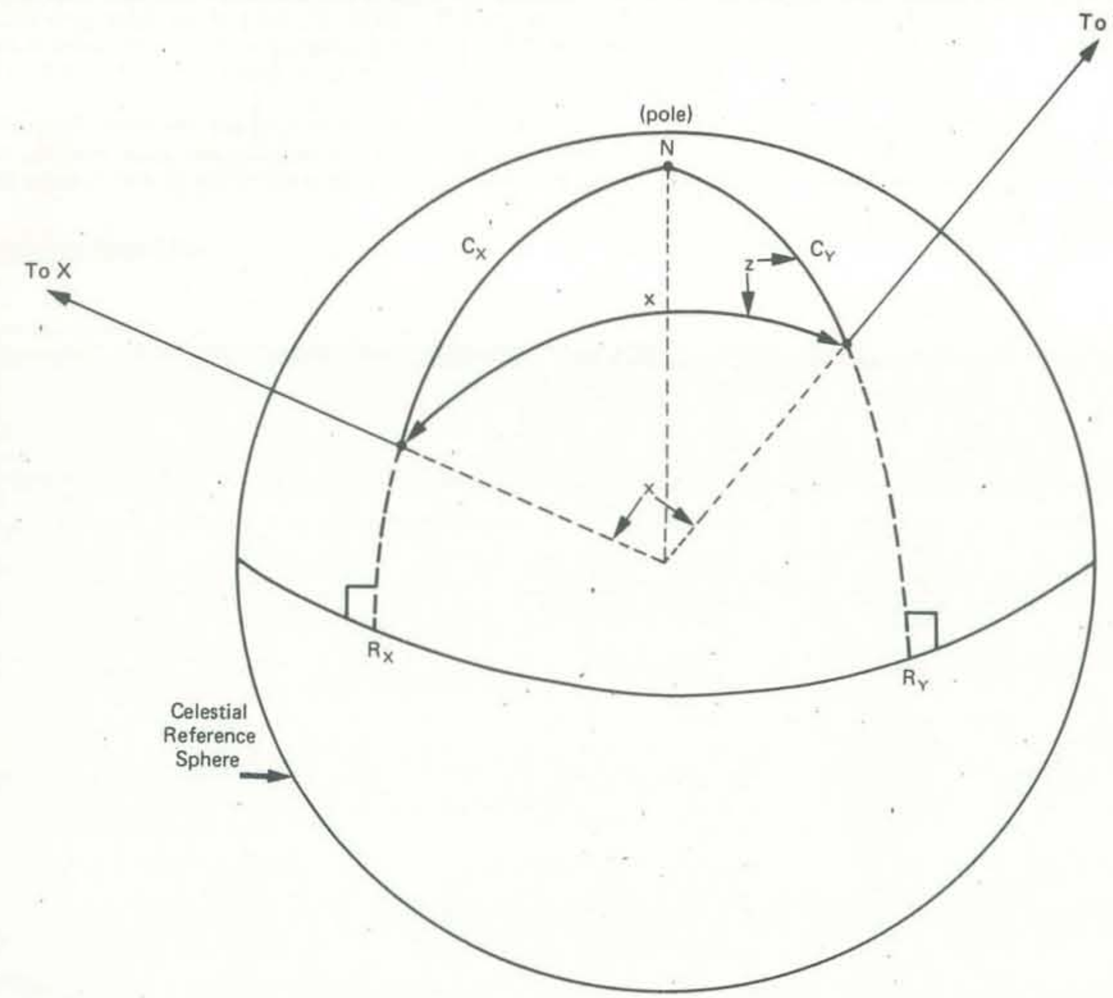
$$l = (d^2 + s^2 - 2ds \cos x)^{1/2}$$

Figure 1. Two Dimensional Analysis

Three-Dimensional Analysis. The two-dimensional analysis is inherent in the three-dimensional analysis: each observed star defines a plane passing through the star and the line of spacecraft motion. Angle x is not given directly but must be determined from the given right ascensions and codeclinations (codeclination, the complement of declination, is the angle of the star from the north celestial pole) of stars X and Y by the law of cosine of spherical trigonometry (Figure 2). Angle y′ is now determined as described above. Angle z (needed later) is determined by the four-parts law of spherical trigonometry.

The three-dimensional problem is concluded by finding the right ascension R'_X and codeclination C'_X of star X in the celestial reference sphere now centered at L (see Figure 3). Angle z, formed by the plane passing through the inertial NS axis of the reference spheres at S and L and the plane of star X and the path SY, is the same angle in both spheres. Again we have three consecutive parts: angles y, z and C_Y (the supplement of codeclination of star Y). Again we use the law of cosines and the four-parts formula of spherical trigonometry, this time to find the codeclination and right ascension (with respect to the reference direction) of star X observed at the spececraft location.

Working on a Small Computer. A small computer can perform a more limited but interesting determination of the manner in which a familiar constellation changes shape as the spacecraft heads toward an arbitrary point in that constellation. Since the stars all diverge from the pole of approach, a simpler and less distorted plot is possible. Angle z is the polar angle and angle y is the radial coordinate, the angular "distance" from the pole. For convenience, relevant data and a sample calculation are given. The constellation is the Big Dipper and the destination point is the center star of the seven stars.



$$\cos x = \cos C_X \cos C_Y + \sin C_X \sin C_Y \cos (R_X - R_Y)$$

where C_X, C_Y, R_1 and R_Y are the codeclinations and right ascensions of X and Y respectively.

$$\tan z = \frac{\sin (R_Y - R_X)}{\dfrac{\sin C_Y}{\tan C_X} - \cos C_Y \cos (R_Y - R_X)}$$

Figure 2. Determination of Angle X

Sample Calculation. The calculation below is for the destination star and the observed end star of the handle at spacecraft distances of 0, 20, 40 and 60 light years from the solar system (63, 43, 23, and 3 light years respectively from the destination star). Visual magnitude is given by the formula: visual magnitude = absolute magnitude − 7.566 − 5 x $\log_{10}$ (distance). The visual magnitude of the destination star as seen from the solar system is 3.3 (dim star). At 20, 40 and 60 light years the visual magnitudes are 2.5, 1.14 and -3.28. Brightness increases with decreasing numerical value of visual magnitude. The system is a carryover from the practice of listing and numbering stars in order of brightness. The last visual magnitude is approximately that of Venus at greatest splendor.

To find the orientation of the end star of the Big Dipper's handle from the destination pole, we have

$$\cos x = 0.8819 \times 0.8401 + 0.4714 \times 0.5424 \times \cos(2.891 - 3.204)$$
$$= 0.984$$
$$x = 10.22$$
$$\sin x = 0.1773$$

$$z = \tan^{-1}\left[\frac{\sin(2.891 - 3.204)}{0.5424 \times \frac{0.8819}{0.4714} - 0.8401 \times \cos(2.891 - 3.204)}\right]$$

$$z = -55.02^\circ$$

Thus, the end star is 10.2° from the destination star and −55.02° (counterclockwise) from the plane through the spacecraft north and the destination star. As we travel toward the star, polar angle −55.02° remains fixed but the observed star moves radially outward from the destination pole. At the first jump of 20 light years we have

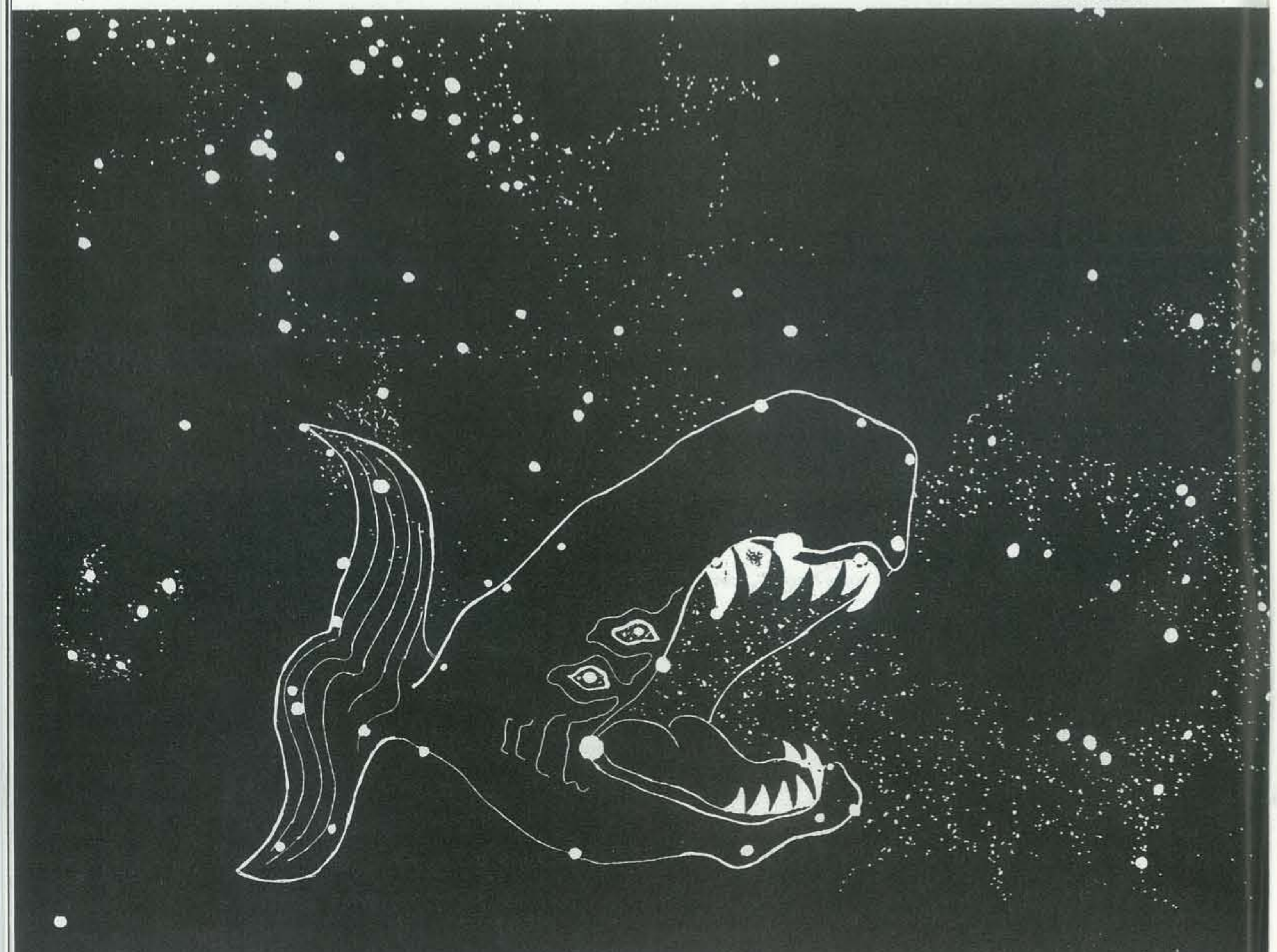
$$y = 180^\circ - \tan^{-1}\left[\frac{\sin 10.22}{\frac{20}{105} - \cos 10.22}\right] = 12.60^\circ$$

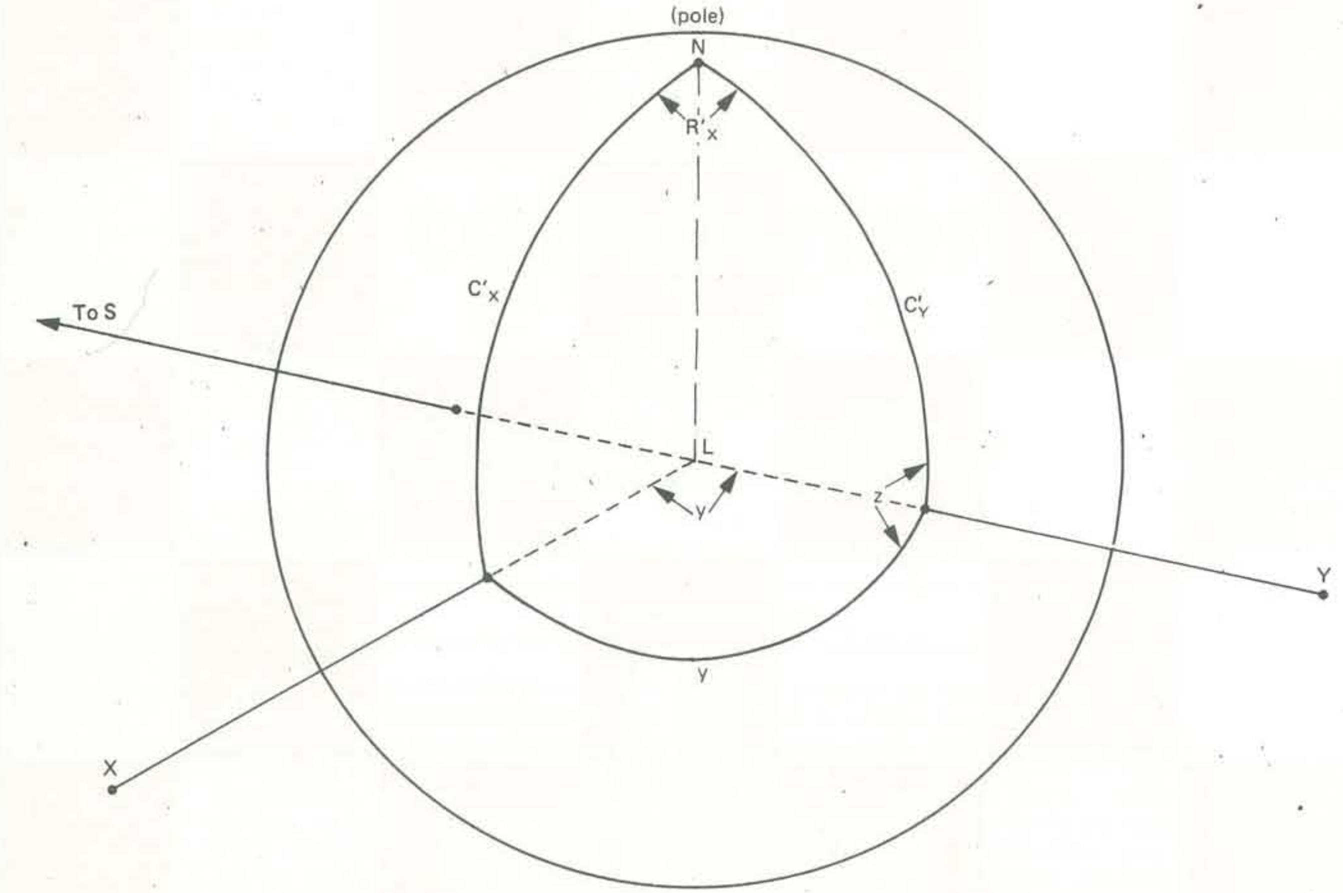
At jumps of 40 and 60 light years from the sun, angle y is 16.38° and 23.25°. At 0, 20, 40 and 60 light years from the sun, the distances and visual magnitudes of the observed star are 105 and 1.84, 85.39 and 1.39, 66.02 and 0.83, and 47.17 and 0.10 respectively.

Performing the calculations for the five other observed stars of the world-renowned constellation and polar plotting the results will yield the Big Dipper as we see it, and in three altered patterns. ■

(Listing on page 26)

The Leviathan of Altair





$$\tan R'_X = \frac{\sin z}{\frac{\sin C'_Y}{\tan y} - \cos C'_Y \cos z}$$

$$\cos C'_X = \cos C'_Y \cos y + \sin C'_Y \sin y \cos z$$

Figure 3. Determination of New Orientation Coordinates of Observed Star

STAR (From end of Bowl)	RIGHT ASCENSION Radians	CODECLINATION Sine	Cosine	DISTANCE Light Years	ABSOLUTE MAGNITUDE (Intrinsic brightness)
1.	2.891	0.4714	0.8819	105	-0.7
2.	2.882	0.5519	0.8339	78	0.5
3.	3.110	0.5904	0.8071	90	0.2
4.	3.204	0.5424	0.8401	63	1.9
5.	3.374	0.5582	0.8297	68	0.2
6.	3.504	0.5731	0.8195	88	0.1
7.	3.608	0.6505	0.7495	210	-2.1

the computer as chess ally

by Mike Gabrielson

A great deal of effort has gone into the creation of computer programs that play chess. As a result, chess enthusiasts have come to think that "computer" is synonymous with "opponent." But nothing precludes using computers in other chess-related ways. Since computers are such general purpose tools, they can be adapted to play many roles, even in chess. In fact, there is a real need to make more computers be our chess allies instead of our chess adversaries.

The Heartbreak of Chess Analysis

You say you're a chess fan who has often spent long nights hunched over a board, with your nose buried in one of the many volumes in your chess library? Quick, then! Answer these:

a) How many possible replies to 1 P-K4 . . . are there for Black which immediately attack White's pawn?
b) What is the strategy behind the Exchange Variation of the Caro-Kann Defense?
c) When can the Noah's Ark Trap appear?
d) What position can the Bishop's Opening transpose into after 3 . . . B-B4?
e) Who, and what move, won the fourth game of Fisher *vs.* Spassky?

(Answers are in sidebar, page 20)

The incredible number and range of possible problems that can be posed (as the above samples demonstrate), give chess its infamous reputation for an ability to absorb infinite hours of study and analysis. Just determining whether or not a problem has ever been tackled before can be a major research effort.

Some chess problems are simple, others seem impossible. Some are concerned with short-lived tactics, others with long-range strategy. Some are mere trivia, others address unanswerable questions mired in the philosophy of chess. But almost all chess problems share at least one interesting trait: they are attacked by chess players using the crudest of tools and age-old methods!

Here is a not-too-unusual scenario: Fred, a strong class B player, has just lost a game at the local coffeehouse tourney. By the tenth move, the game had developed into play that Fred had never seen before. Anxious to replay and analyze his loss, and curious to know where the game diverged from known play, Fred decides to spend a few hours with his books and boards. He pulls out the Encyclopedia of Chess Openings (ECO) and three other large tomes from his overflowing bookshelves. He sets up two boards (one for actual play-by-play, another for investigating variations of the game so as not to disturb the position of the actual line played), and then begins his analysis.

While studying the fifteenth move, Fred begins to vaguely recall a description of a similar game (perhaps in a book by Reinfeld?), but a quick search of his library finds nothing. Fred does discover some interesting notes he made in the margin of the ECO a year or so ago, but they are so cryptic they now prove undecipherable. A couple of hours later, Fred is satisfied he now knows where he lost the game, and concludes his analysis by stuffing two pages of new and hastily written notes inside the jacket of the book he found most helpful.

Mike Gabrielson – Box 2692, Stanford, CA 94305.

The moral of the story can be expressed as an equation:

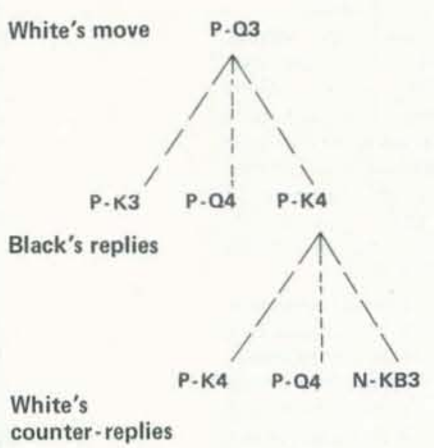
unwieldy boards
+ forgetful brains
+ incompatible books
= tremedous overhead,

which acts as an effective barrier for chess players trying to concentrate on the problem at hand. Much of the work of chess analysis is caused by the ponderous techniques. There must be a better way!

No More Books

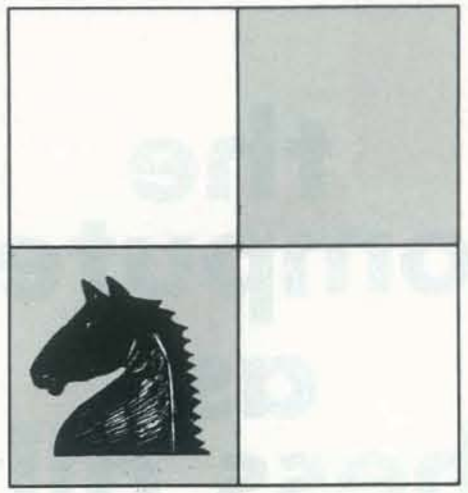
The learning, playing, and studying of chess involves a large amount of information processing. Players are continually gathering, storing, recalling, extracting, updating, classifying, indexing, and cross-referencing their chess knowledge. The sheer volume of information forces the use of books and other permanent records.

But chess information is often best organized as upside-down trees, reflecting the manner in which players explore the game:



A branch down the tree is selected, variations on the move are investigated and then possibly abandoned as a retreat is made up the tree of moves, until a more promising branch back down the tree is found. The printed page is a poor medium for documenting this type of exploration. Books are more ideally suited to simple sequential steps (A then B then C and so on) while chess games consist of steps that continually divide into an explosion of numerous alternatives (A then B or C or D, if B then E or F, if C then G or H or back to B, and so on). It's amazing how much jumping back and forth between pages one must do while using a book of chess openings.

A good player's chess experience is constantly growing and expanding. Another serious drawback of current chess tools is the inability of books and manuals to grow with the reader and to aid in the pooling of knowledge. Today, most players like Fred rely on margin notes, or try to develop their own system (usually on paper) for cataloging the



chess data important to them. Unfortunately, most paper-oriented chess notation systems developed by players are just as clumsy and limited as the books they try to make up for.

For a long time, computers have been used to manage information that is too voluminous or awkward for manual methods to deal with. It is no less true that the computer can similarly be a tool for the student of chess. The computer can provide support in the three major areas where current chess analysis tools are lacking, namely:

- Continual graphic feedback on the board position of interest (instead of the occasional pictures sprinkled throughout most chess books),
- Procedures for easy exploration of games, positions, and variations, in a manner suited to the player's thought processes (instead of the player having to follow a presentation already mapped out on paper),
- A virtually unlimited memory (for storing a player's ideas, annotations, and discoveries) that grows with the player, while maintaining consistency and organization (instead of an *ad hoc* hodge-podge of incoherently scribbled notes).

An Actual Implementation

COCO is a computer program designed to aid in the analysis of chess games. COCO is an acronym for Catalog Of Chess Openings, but this name was chosen for its convenient abbreviation, not because it is terribly accurate in describing the program's capabilities.

COCO communicates with the operator (the person operating COCO) via a 24 by 80 character display screen and a keyboard. A disk is used for storing data. (A display, keyboard and disk are standard equipment on most computers today – see sidebar.) COCO starts by generating the display shown in Figure 1.

The display is dominated by a chess board which occupies the left half of the screen. The board always shows the current position under study. Pieces are denoted by Q for Queen, K for King, and so on, with a prefix of B for Black and W for White.

Most of the right half of the screen is occupied by a twenty-line area reserved for the moves and annotations the operator wishes to have displayed alongside the current board position. This is the most important area of the screen: it serves as a "scratchpad" where the operator creates, changes deletes and otherwise manipulates any information important at that point in the game under study.

Each of the twenty scratchpad lines is labeled by a number along the left edge. These are *not* move numbers, but merely allow the operator to refer to a line on the screen when typing input at the keyboard. For example, to change some information near the center of the scratchpad, the operator might request COCO to allow the replacement of text on line 10.

The twenty lines are split into two columns. The leftmost nine character positions of each line are reserved for the description of a legal move on the current board. The remaining character positions up to the right edge of each line may contain arbitrary text. These moves and comments are supplied by the operator, but once entered at the keyboard, COCO will remember the moves and comments along with the board position, until they are later changed or deleted by the operator. The tops of these two columns are labeled. "Black" or "White" is above the move column, to indicate which player is allowed to select from the given moves for that specific board, while "Comment" permanently heads the text side. A move does not have to appear on every line. In Figure 2 we see that the operator has chosen to save only two moves for consideration by Black at that stage in the game (one move is on line one; the other, on line four).

The line in the bottom right corner of the display is what allows the operator to actually manipulate the scratchpad and to put COCO through its paces. The line always contains seven cryptic reminders showing the syntax of the seven possible commands that the operator may type in at the keyboard. The continual prompt "Command?" precedes the cursor (a blinking underline). This is where operator input from the keyboard is initially echoed on the display. (When the operator types a key, the character typed is also displayed at the cursor position.) By typing just a few characters, the operator can compose a command that requests COCO to perform some useful action, which actually occurs when the operator hits the RETURN key after typing the command. When finished executing the command (which often causes the display to change in some way), COCO will reprompt and wait for another command.

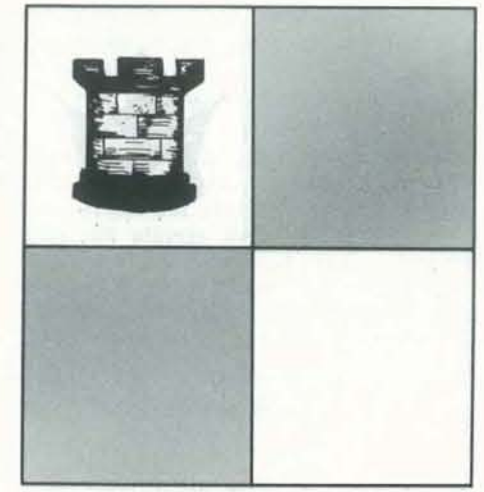
The Seven Commands

COCO understands seven different commands which the operator can type in at the keyboard to 1) add a move to the current list being displayed, 2) subtract a move from the list, 3) add a comment to a line, 4) make a move, 5) go back to a previous position, 6) forget a board position, and 7) quit the session. All of the commands are one, two or three typed characters, in the following format ("#" denotes a one- or two-digit scratchpad line number):

Format	Command
+#	Add a move.
-#	Subtract a move.
C#	Comment a line.
#	Make a move.
B	Go back a move.
-	Forget a board.
Q	Quit.

Adding a move: The operator can instruct COCO to add a move to the list of moves being displayed. This is done by typing a plus sign (+) followed by a scratchpad line number. For example, "+12" indicates the operator wants to add a new move to the list being displayed and that the move will be shown on scratchpad line twelve. After the operator types "+12" and hits the RETURN key, COCO will move the blinking cursor to line twelve in the scratchpad and wait for the operator to type the actual move. The operator does this in two steps. First, he must type a four-character, algebraic description of the move. For example, "G1F3" as a first move means "knight to king bishop three." With COCO, algebraic moves are always typed from White's point of view (i.e., "A1" is the lower left square on the screen, regardless of which side has the move). Algebraic moves are always four characters, to keep them in a simple form understandable by COCO, since it is the algebraic representation of each move that COCO uses to actually manipulate pieces on the board. Once the operator has typed the four character algebraic move and hit the RETURN key, COCO will then wait for the operator to type in the descriptive form of the move. This can be anything up to nine characters, such as "N-KB3." The descriptive form for each move is what is actually displayed in scratchpad's left column.

Once the move has been entered in algebraic and descriptive form, COCO will always display that move whenever the board position (where the move was typed in) is reached. And any move displayed can then be made by COCO if so commanded by the operator (as described below) in order to reach a subsequent board position. COCO uses the comput-



er's disk memory for storing moves, comments and board positions. Since disks can contain large amounts of data in a very small amount of space, COCO is typically able to store much more information than one operator will ever want to generate or access.

Note that if a move already exists on the scratchpad line where the new move is being added, then the old move is "forgotten."

Subtracting a move: Once a move has been added by the operator it can also be deleted from the list. This is done by typing a minus sign (-) followed by a scratchpad line number. For example, "-7" is the command to COCO to "forget" the move, if any, currently displayed on scratchpad line seven.

Once a move is deleted in this manner, it is no longer displayed with the board, and the move cannot be used to reach a subsequent board position. Of course, by using the "+#" command form, a new (or even the same) move can later be placed back in the same scratchpad position.

Commenting a line: It is frequently desirable to place arbitrary comments alongside the moves in the scratchpad area. This is done by typing "C" followed by a scratchpad line number. For example, "C3" indicates the operator wants to create or change the comment on scratchpad line three. After the operator types "C3" and hits RETURN, COCO will position the cursor to line three in the scratchpad (to the right of the move column) and wait for the operator to type in a comment. A comment is anything up to 25 characters and ends when the operator hits RETURN. Typing only the RETURN generates a "null" comment, useful for erasing any previous comment already on the scratchpad line. Comments may be placed on scratchpad lines that have no moves.

Like moves, comments are remembered by COCO and displayed for each board.

The commands described so far simply let the operator create or change data in the scratchpad. The following commands make COCO use the data put there.

Making a move: Once a move is displayed by COCO, the operator can have COCO carry out that move in order to display the resulting board position. This is done simply by typing a scratchpad line number. For example, typing "4" and hitting RETURN instructs COCO to execute the move currently displayed on scratchpad line four. COCO will display the new board position reached by making that move. If the resulting board position had moves and comments saved with it during a previous visit, then of course COCO will recall and redisplay the moves and comments now that the board position has again been reached.

When a move is made and the chosen move has a comment, then the comment is displayed above the scratchpad as a reminder to the operator. And as mentioned previously, the color of the side having the next move is changed at the top of the moves column whenever a move is made.

Going back a move: At any time other than when the starting board position is displayed, the operator may ask COCO to go back to the board position which immediately preceded the current one being displayed. This is done by typing "B", after which COCO will change the display to show the board position previously in effect. As usual, any moves and comments associated with the board will be displayed also.

Forgetting a board: At any time (other than when the starting board position is displayed), the operator may ask COCO to go back to the board position which immediately preceded the current one and to also "forget" all moves and comments associated with the board before taking the step back. This is done by typing a minus sign (-). The effect seen by the operator is the same as when the "B" command is given, except that if moves are made such that the forgotten board position is once again reached, the scratchpad will be blank, since COCO erased all information from its memory about the board when the "-" command was given.

Quitting: When finished working with COCO and it's time to quit, the operator should give the "Q" command. (This requirement is actually dependent upon the type of computer COCO happens to be programmed on.)

How COCO Helps

COCO is a useful and convenient medium for storing, exploring and analyzing large amounts of chess data. COCO helps by:

- Continually displaying the current board, moves and comments under study.

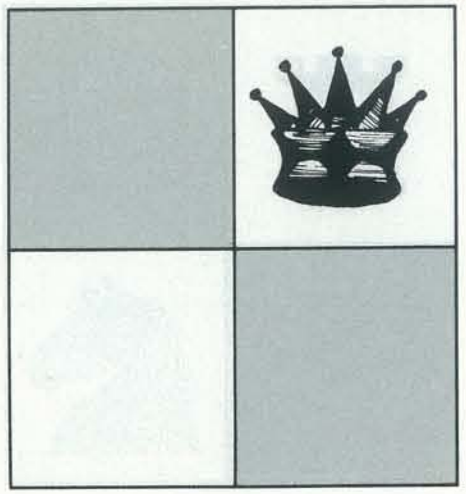
- Letting the operator explore and browse the data quickly and easily, including backing up when necessary.
- Most importantly, providing a method for saving a player's comments and annotations with any and every board position, in an organized and consistent, yet expandable, manner. This last feature is what makes COCO so different and so much more powerful than books and boards.

Anything that can be said or written about a chess game or position can be stored and remembered by COCO, then later displayed, analyzed and even modified. COCO's ability to grow and continually record the explorations of the operator make it a powerful tool for any chess enthusiast.

Improvements You Can Make

The version of COCO just described is only a prototype, not a proposal of some ultimate method. Anyone who uses COCO for just a few minutes can think of many desirable improvements that would make the program an even better tool.

With the current version of COCO, the scratchpad is the most limited and



underdeveloped part of the program. The current simple implementation is surprisingly powerful, but a truly useable version would allow any number of lines of moves and comments to be stored with each board and accessed in the scratchpad.

The scratchpad is essentially an area for text editing. Text editing is a computer application that has received a lot of attention from programmers, and many sophisticated, display-oriented text editing programs have been developed which allow an operator to modify large blocks of text through the limited "window" available on computer displays. By putting the same capability in COCO, an arbitrarily large number of lines of moves and comments could be saved with each board position, giving the operator complete freedom to create and decide on the amount of information to be stored for each turn.

COCO could also use a better board display. Simple character mnemonics, such as "BQ" for Black Queen, allow quick implementation of COCO's current board, but much more sophisticated graphic capabilities are available for many computer terminals, so that a more aesthetically pleasing board is possible. In fact, many existing chess playing programs generate displays showing pieces as easy-to-recognize figures.

Entering moves in both algebraic and descriptive notation is cumbersome. With better programming, COCO would only need input in one form. With better hardware, the operator wouldn't even have to type in moves. Light pens or touch-sensitive displays would allow the opera-

Computer Anatomy for Beginners

Physically internal to a computer are its high-speed memory and central processing unit, but these are usually small, enclosed in some type of cabinet, and operate just fine without a lot of handling by humans. On the other hand, externally connected peripherals such as displays, keyboards and disks are designed for human interaction and handling (although this is somewhat less true for disks).

The *display screen* is usually a television-type picture tube, often called CRT, for Cathode Ray Tube, and can display numbers, letters and other characters. Some computer displays are capable of more advanced graphics, such as lines, bars, colors, curves, and so on. The main method the computer uses to output results to the operator is to display them on the screen. Many computers also (or instead) use printers which type results on paper. Displays are faster and save paper, but don't generate permanent hard copies like printers do.

The *keyboard*, which is manipulated just like a typewriter keyboard, is the main method the operator uses to input data to the computer. More exotic alternatives are available, such as light pens, touch-sensitive displays and voice-recognition units, but keyboards are by far the most often used of input devices.

Disks are a place for the computer to store programs and data. Disks serve as a slower but larger, external memory in addition to the computer's faster and smaller internal memory. A disk is a sort of cross between a record player and a tape recorder – it uses a mixture of the two technologies. A flat, round plastic or metal "platter" is coated with material that can be magnetized. Information is stored on the platter in much the same way information is stored on tape by a tape recorder: a small "head" rests on or near the surface of the disk and is used to magnetically change the information recorded there. A motor and hub are used to spin the disk around. At the same time, the head is attached to an arm that can move back and forth over the spinning disk. This is similar to a record player, since the movement of the disk and the arm allows the head to be positioned over any spot on the disk, almost like a record player needle can be positioned over any spot on a record. On a record, a single groove is actually cut into the plastic surface for recording information like music, and slowly spirals in toward the center. On a disk, information is magnetically recorded (the surface stays smooth) in concentric circles called tracks.

There is a tremendous variety of disk types and sizes, and the technology is always improving. Floppy disks (a flexible plastic platter) are at the smaller, slower and cheaper end of the scale, and normally hold about 70,000 to 250,000 or more characters of data. Larger, faster and more expensive rigid disks (metal platters, often with fixed, multiple heads for quick access to data) can now hold up to 500,000,000 characters of data and more.

Answers to heartbreak questions:

a) Three.

b) White will exchange with Black's well-posted Queen Bishop to increase a lead in development and to gain time for castling on the Queen side.

c) In the Ruy Lopez, the Morphy Defense with 5 . . . B-K2 might make White fall for the Trap on the seventh move:

6 P-Q4 PxP
7 R-K1 P-QN4
8 B-N3 P-Q3
9 NxP?? NxN
10 QxN P-B4
11 . . . P-B5 winning a Bishop.

d) 4 P-B4 transposes into the King's Gambit Declined.

e) Neither player won. The 45th move ended the game as the series' first draw.

tor to simply point at the pieces to be moved. Voice input is another possibility (see box). The ability to start COCO at any board configuration would be helpful, too, even though it only takes the operator a few moments to make COCO step through the moves to reach a midgame position.

Note that COCO's current program does essentially no checking for invalid moves or even special moves (such as castling or *en passant* captures).

One last improvement worth mentioning would be the standardization of a method for storing chess data on disk, so that operators could transfer or merge the data bases from separate computers.

Final Notes

For a large number of occasional players, chess is a some-time game. For many serious fans, chess is a way of life. And for some hard-core fanatics, chess IS life. Regardless of their degree of enthusiasm, all players have shared the same tools while learning and playing: boards, books and brains. Now the computer can be added to that list, and players can be freed from the tedious and mundane aspects of the study of chess. Computer-aided game analysis has arrived! ■

(LISTING ON PAGE 22)

Editor's note: The author wrote COCO in Data/BASIC, which is specifically for Microdata computers (see Listing on page 22). It will be a rewarding experience for chess players and computer hobbyists alike to re-write (and upgrade!) COCO in a language which will run on your machine. The result will be something far more valuable than a computer chess opponent... it will be a computer chess ally.

FIGURE 1.

```
*****     *****     *****     *****       COCO                12:45:48  04 JAN 1981
*B*R* B N *B*B* B Q *B*K* B B *B*N* B R
*****     *****     *****     *****          White    Comment
     *****     *****     *****     *****  1 P-K4     King's pawn openings
 B P *B*P* B P *B*P* B P *B*P* B P *B*P*  2 P-Q4     Queen's pawn openings
     *****     *****     *****     *****  3 P-KB4    Bird's opening
*****     *****     *****     *****       4 P-QB4    English opening
*****     *****     *****     *****       5 N-KB3    Reti opening
*****     *****     *****     *****       6 N-QB3    Dunst opening
     *****     *****     *****     *****  7 N-KR3    Paris opening
     *****     *****     *****     *****  8 N-QR3    Durkin's attack
     *****     *****     *****     *****  9 P-KN4    The "Spike"
*****     *****     *****     *****      10 P-QN4    Polish opening
*****     *****     *****     *****      11 P-QB3    Saragossa opening
*****     *****     *****     *****      12 P-K3     Van't Kruys opening
     *****     *****     *****     ***** 13 P-QR3    Anderssen's opening
     *****     *****     *****     ***** 14
     *****     *****     *****     ***** 15
*****     *****     *****     *****      16
*W*P* W P *W*P* W P *W*P* W P *W*P* W P  17
*****     *****     *****     *****      18
     *****     *****     *****     ***** 19
 W R *W*N* W B *W*Q* W K *W*B* W N *W*R* 20
     *****     *****     *****     ***** +#  -  -#  #  C#  B  Q  Command?
```

FIGURE 2.

```
*****     *****     *****     *****       COCO                12:49:28  04 JAN 1981
*B*R* B N *B*B* B Q *B*K* B B ***** B R   Blumenfeld counter gambit
*****     *****     *****     *****          Black    Comment
     *****     *****     *****     *****  1 PxP      Transposes into the
 B P *B*P*     *B*P*     *B*P* B P *B*P*  2          Benoni line.
     *****     *****     *****     *****  3
*****     *****     *****     *****       4 P-QN4?!  Risky! (Unnecessary too,
*****     *****     *B*P* B N *****       5          considering the above
*****     *****     *****     *****       6          alternative.)  This is
     *****     *****     *****     *****  7          hoping for
     ***** B P *W*P*     *****     *****  8
     *****     *****     *****     *****  9            QPxP  BPxP
*****     *****     *****     *****      10            PxP   P-Q4
*****     *W*P*     *****     *****      11
*****     *****     *****     *****      12          giving a strong center
     *****     *****     *****     ***** 13          and open lines for an
     *****     *****     *W*N*     ***** 14          expendable pawn.  But
     *****     *****     *****     ***** 15          watch out for
*****     *****     *****     *****      16
*W*P* W P *****     *W*P* W P *W*P* W P  17            B-N5!
*****     *****     *****     *****      18
     *****     *****     *****     ***** 19
 W R *W*N* W B *W*Q* W K *W*B*     *W*R* 20
     *****     *****     *****     ***** +#  - | -#  #  C#  B  Q  Command?
```

The Computer as Chess Ally

```
COCO
* COCO - Catalog Of Chess Openings
* Mike Gabrielson - 3/22/79
*
* Copyright (c) 1979 - All Rights Reserved
*
* This program is written in Data/Basic (tm) for the Microdata line of computers.
* Consult Microdata's documentation for information about Data/Basic.
*
* Item structure:     0 Board id
*                     1 Previous board id
*                     2 Next algebraic moves
*                     3 Next descriptive moves
*                     4 Comments
*
OPEN "BOARDS" ELSE PRINT "Chess boards are missing!" ; STOP
PROMPT ""
*
BOARD.ID = "002345643211111111000000000000000000000000000000007777777789ABCA98"
LAST.ID = ""
LAST.COMMENT = ""
*
1 PRINT CHAR(12)
PRINT @(41,0):"COCO":@(59):TIMEDATE():
PRINT @(41,1):LAST.COMMENT:
*
FIRSTCOL = 0
FOR RANK = 0 TO 21 STEP 3
  LASTCOL = 30 + FIRSTCOL
  FOR ROW = 0 TO 2
    FOR COL = FIRSTCOL TO LASTCOL STEP 10
    PRINT @(COL, RANK+ROW):"*****":
    NEXT COL
  NEXT ROW
  FIRSTCOL = 5 - FIRSTCOL
NEXT RANK
*
PRINT @(41,23):"+#  -  -#  #  C#  B  Q":
READ BOARD FROM ICONV(BOARD.ID,"MX") ELSE BOARD = ""
IF LAST.ID # "" THEN
  BOARD<1> = LAST.ID
  WRITE BOARD ON ICONV(BOARD.ID,"MX")
  END ELSE LAST.ID = BOARD<1>
*
PRINT @(44,2):
IF BOARD.ID[1,2] = "01" THEN PRINT "Black": ELSE PRINT "White":
PRINT @(53):"Comment":
*
FOR I = 3 TO 66
IF MOD(I,2) = MOD(INT((I-3)/8)+1,2) THEN COLOR = "*" ELSE COLOR = " "
PRINT @(MOD(I-3,8)*5+1,INT((I-3)/8)*3+1):
ON SEQ(ICONV(BOARD.ID[I,1],"MX"))+1 GOTO 10,20,30,40,50,60,70,80,90,100,110,120,130
10 GO TO 140
20 PRINT "B":COLOR:"P":
GO TO 140
30 PRINT "B":COLOR:"R":
GO TO 140
40 PRINT "B":COLOR:"N":
GO TO 140
50 PRINT "B":COLOR:"B":
GO TO 140
60 PRINT "B":COLOR:"Q":
GO TO 140
70 PRINT "B":COLOR:"K":
GO TO 140
80 PRINT "W":COLOR:"P":
GO TO 140
90 PRINT "W":COLOR:"R":
GO TO 140
100 PRINT "W":COLOR:"N":
GO TO 140
110 PRINT "W":COLOR:"B":
GO TO 140
120 PRINT "W":COLOR:"Q":
GO TO 140
130 PRINT "W":COLOR:"K":
140 NEXT I
*
FOR LINE = 1 TO 20
  PRINT @(41,LINE+2):LINE "R#2":" ":BOARD<3,LINE> "L#9":BOARD<4,LINE>
NEXT LINE
*
PRINT @(65,23):"Command?":
INPUT COMMAND,3:+
CHAR = COMMAND[1,1]
LINE = COMMAND[2,2]
*
BEGIN CASE
CASE CHAR = "Q"
  STOP
*
CASE (CHAR = "B") AND (LAST.ID # "")
  BOARD.ID = LAST.ID
  LAST.ID = ""
  LAST.COMMENT = ""
*
CASE CHAR = "-"
  IF (LINE = "") AND (LAST.ID # "") THEN
    DELETE ICONV(BOARD.ID,"MX")
    BOARD.ID = LAST.ID
    LAST.ID = ""
    LAST.COMMENT = ""
    END ELSE IF (LINE MATCHES "1N") OR (LINE MATCHES "2N") THEN
             BOARD<2,LINE> = ""
             BOARD<3,LINE> = ""
             WRITE BOARD ON ICONV(BOARD.ID,"MX")
             END
*
CASE CHAR = "C"
  IF (LINE MATCHES "1N") OR (LINE MATCHES "2N") THEN
    PRINT @(53,LINE+2):SPACE(25):@(53):
    INPUT COMMENT,25:+
    BOARD<4,LINE> = COMMENT
    WRITE BOARD ON ICONV(BOARD.ID,"MX")
    END
*
CASE CHAR = "+"
  IF (LINE MATCHES "1N") OR (LINE MATCHES "2N") THEN
    PRINT @(44,LINE+2):SPACE(9):@(44):
    INPUT ALGEBRAIC,4:+
    PRINT @(44):SPACE(4):@(44):
    INPUT DESCRIPTIVE,9:+
    BOARD<2,LINE> = ALGEBRAIC
    BOARD<3,LINE> = DESCRIPTIVE
    WRITE BOARD ON ICONV(BOARD.ID,"MX")
    END
*
CASE CHAR MATCHES "1N"
  LINE = COMMAND
  IF (LINE MATCHES "1N") OR (LINE MATCHES "2N") THEN
    ALGEBRAIC = BOARD<2,LINE>
    SRC = (8-(SEQ(ALGEBRAIC[2,1])-48))*8 + SEQ(ALGEBRAIC[1,1]) - 64 + 2
    DST = (8-(SEQ(ALGEBRAIC[4,1])-48))*8 + SEQ(ALGEBRAIC[3,1]) - 64 + 2
    PIECE = BOARD.ID[SRC,1]
    LAST.ID = BOARD.ID
    BOARD.ID = BOARD.ID[1,SRC-1]:"0":BOARD.ID[SRC+1,LEN(BOARD.ID)-SRC]
    BOARD.ID = BOARD.ID[1,DST-1]:PIECE:BOARD.ID[DST+1,LEN(BOARD.ID)-DST]
    BOARD.ID = "0":(1-BOARD.ID[2,1]):BOARD.ID[3,LEN(BOARD.ID)-2]
    NEW.COMMENT = BOARD<4,LINE>
    IF NEW.COMMENT # "" THEN LAST.COMMENT = NEW.COMMENT
    END
END CASE
*
GO TO 1
*
END
```

Recreational COMPUTING Back Issues

PCC NEWSPAPER

HALF OFF SALE!

All 12 Available back issues of PEOPLE'S COMPUTER COMPANY NEWSPAPER for only $12.

- #3 **May-June 1973** ZINGO/"Going to Bid"/ How to Buy an Edusystem/BASIC MUSIC CHOMP. $2.00
- #10 **July-Aug 1974** A Fantasy of Future Forms/Low-Cost Home School Microprocessor Systems/INCHWORM/MAZE/ Analysis of Teaser/Computer Illiteracy Problem by Peter Lykos. $2.00
- #13 **Jan-Feb 1975** Build your own BASIC/ Mysterious RND/More INCHWORM/ Computers & Art/BIOSIN Son of Hardware. $2.00
- #18 **Nov-Dec 1975** NUMBER/Tiny BASIC/ Analog to Digital Conversion/Minuet in G Major. $2.00
- #19 **Jan-Feb 1976** Space Games/Tiny BASIC Soloworks/DIDDLE/STTR/More Music. $2.00
- #20 **Mar-Apr 1976** Games for the Pocket Calculator/SQUARES/Make Believe Computers/STRTRK/Huntington Computer Project. $2.00
- #21 **May-June 1976** Biofeedback/POUNCE/ "We're Building Our Own Computer"/ Programmer's Toolbox/CCC/TV as Terminal and Game Center. $2.00
- #22 **July-Aug 1976** Low-Cost Software/Tiny BASIC, Tiny Trek/Your Brain is a Hologram. $2.00
- #23 **Sept-Oct 1976** Dungeons & Dragons/ HATS/One on One/PLANETS/The Positive of Power Thinking. $2.00
- #24 **Nov-Dec 1976** STORY/SNAKE/More Build Your Own Computer/Introducing PILOT/FROGS. $2.00
- #25 **Jan-Feb 1977** REVERSE/Robots/Tiny PILOT/Space & Computers/Conversational Programming. $2.00
- #26 **Mar-Apr 1977** Z-80 PILOT/6502 Assembly Programming/Tiny BASIC for Beginners. $2.00

PEOPLE'S COMPUTERS

- #27 **May-June 1977** Graphics, Computers & Copyright Laws, PILOT & Tiny BASIC, Home Computers for Beginners, Women & Computers. $2.50
- #28 **July-Aug 1977** Heathkit's 8080 & LSI-11, Do-It-Yourself CAI, Robot PETs. $2.50
- #29 **Sept-Oct 1977** Computer Networks, The $595 PET, More Tiny Languages. $2.50
- #30 **Nov-Dec 1977** Oz Graphics, Biofeedback & Micros, Our PET's First Steps. $2.50
- #31 **Jan-Feb 1978** Robots, Video Discs, PET Update & Programs, PASCAL & COMAL. $2.50
- #32 **Mar-Apr 1978** Epic Computer Games, TRS-80 Review, Micros for the Handicapped. $2.50
- #34 **July-Aug 1978** Computer Whiz Kids, Public Access to Computers, Man-made Minds. $2.50
- #35 **Sept-Oct 1978** Computers & Museums, Kingdom, APL, Sorcerer of Exidy. $2.50
- #36 **Nov-Dec 1978** The Return of the Dragons, APL, Animated Games for TRS-80. $3.00

RECREATIONAL COMPUTING

- #37 **Jan-Feb 1979** Artificial Intelligence, A fantasy by Jules Verne, TRS-80 Personal Software, The Apple Corps is with Us. $3.00
- #38 **Mar-Apr 1979** Artificial Intelligence, Calculator Comics, Chess Reconsidered, Lord of the Rings. $3.00
- #39 **May-June 1979** Now Peter Can Read, Game of Life, TRS-80 Programs, Detective Game Program. $3.00
- #40 **July-Aug 1979** ComputerTown, USA!, Cryptarithms, DP Games Playing, FORTE Music Programming Language. $3.00
- #42 **Nov-Dec 1979** Shogi Games, Creating New Characters on the Ti 99/4, Music for Atari, TRS-80 & PET. $3.00
- #43 **Jan-Feb 1980** Computing for Health and Equality (PET), Ti Graphics & Animation Pt. 2, More Music for Atari. $3.00
- #44 **Mar-Apr 1980** Special Games Issue! Recreation Apple II Hi-Res Graphics, Galaxy II, TRS-80 Tiny Trek. $3.00
- #45 **May-June 1980** Intro to Computer Music, CBBS Phone Numbers, 6502 Machine Language, Five New Personal Computers. $3.00
- #46 **July-Aug 1980** Fantasy Games! Write Your Own, Wizard's Castle, Star Trek, Wonderful World of Eamon, The End of Fortran Man. $3.50
- #47 **Sept-Oct 1980** 8th Anniversary Issue! Best of PCC, Probability Trees, Computers in Sports Medicine, Wired by Mark Singer. $3.50
- #48 **Nov-Dec 1980** Computerized Voting, Hot Rod Computers and Model T Fords, The Story of Computer-Using Educators, Games! $3.50
- #49 **Jan-Feb 1981** Education Revolution, Interview with Harold Cohen, Micros in China, Composing Music Through Computers. $3.50

Only the issues listed are available. Outside the U.S., add $.50 per issue ordered. Price includes issue, handling and shipment by second class or foreign surface mail. Within the U.S., please allow 6-9 weeks to process your order second class. For faster service within the U.S., we'll ship UPS if you *add* $1.00 for 1-2 issues and $.50 for *each* issue thereafter. We need a street address, not a P.O. Box. Outside the U.S., *add* $1.50 *per* issue requested for airmail shipment.

Please send me the issue(s) circled.

3 10 13 18 19 20 21 22 23 24 25 26 27 28 29 30
31 32 34 35 36 37 38 39 40 42 43 44 45 46 47 48 49

☐ I'm ordering $20 or more and I want a 10% discount!

I enclose $ _________ (U.S. check or money order). Outside the U.S., add $.50 per issue. Please send in this form or facsimile to: Recreational Computing, 1263 El Camino Real, Box E, Menlo Park, CA 94025. This offer expires June 30, 1981.

NAME ______________________

ADDRESS ______________________

STATE ______________________ ZIP ______________________

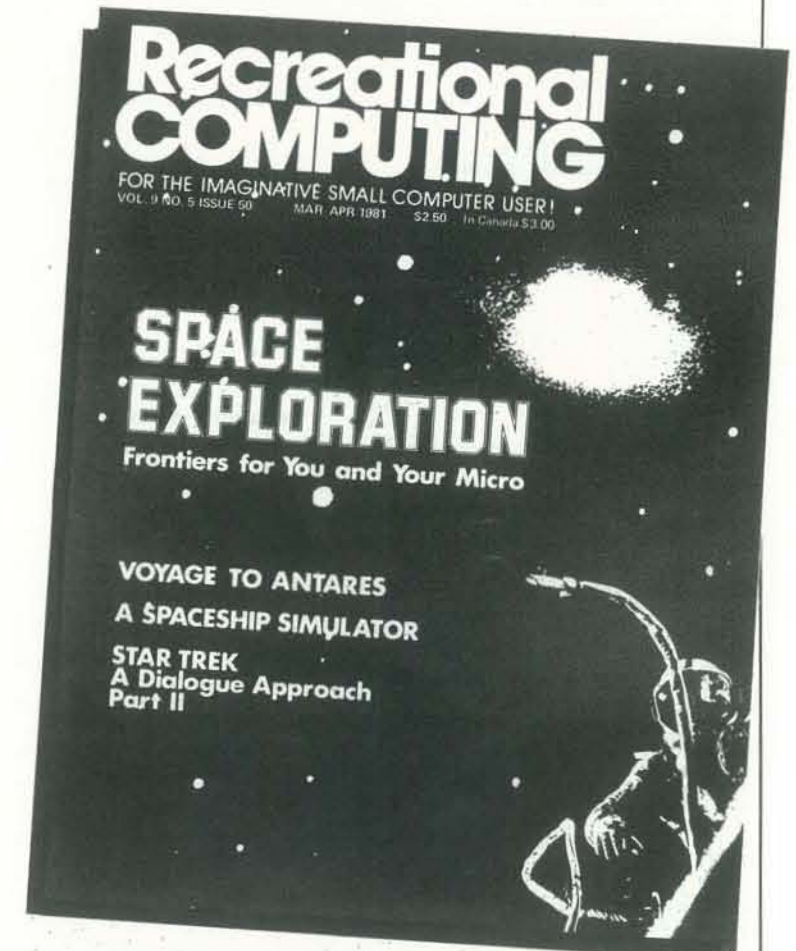
Computers and Learning...

Games and the Arts...

Kids Who Compute...

Programming Problems and Solutions...

Adventure...



All the exciting topics you can imagine fill each issue of RECREATIONAL COMPUTING. • How can your computer help you and your family? • What is the best software to buy? • Is the newest hardware always the best? • How can you better use your computing power? **Our pages answer these questions and more!**

RECREATIONAL COMPUTING is a magazine where learning is fun, and where recreation becomes educational. Our style is easy to read, and makes the important topics easy to understand. Join us in exploring the pleasures and uses of small computers. **Subscribe now** and learn why we were the **first** personal computing magazine, and growing faster than ever!

Subscribe Now!

Yes! I want to subscribe to Recreational Computing!
Please bill me $12 for 6 issues.

Name ______________________

Address ______________________

City, State, Zip ______________________

Dept. M4, Box E, Menlo Park, CA 94025.

Voyage to Antares *(continued from page 15)*

```
10 REM RFD4 - BYBYBAYBEE - MAIN PROGRAM: SKY AT FOUR S/C DISTANCES -
20 REM DAT/TIM: 15AUG78/ 8:54PM
30 DIM GS[60],HS[60],US[60],VS[60],YS[60],ZS[60],TS[3,4]
40 DIM AS[60],BS[60],CS[60],DS[60],ES[60],FS[60],L[4]
50 X=180/PI
60 P=PI/2
70 K=0
80 MAT READ L
85 DATA 10,50,200,500
90 MAT PRINT L
100 PRINT
120 REM
130 REM SELECT DESTINATION
140 REM N=0: NEW DESTINATION
150 REM
160 DISP "NEW DESTINATION INPUT";
170 INPUT N
180 IF N=0 THEN 380
190 REM
200 REM NEW DESTINATION
210 REM
220 DISP "DEC (DEG)";
230 INPUT J
240 DISP "DEC (MIN)";
250 INPUT K
260 C2=J+K/60
270 C=P-C2/X
280 DISP "RA (HRS)";
290 INPUT U
300 DISP "RA (MIN)";
310 INPUT V
320 E1=(U+V/60)*15
330 R=E1/X
340 GOTO 690
350 REM
360 REM OLD DESTINATION
370 REM
380 MAT READ T
390 DATA 16,28.2,-26,-23,0,46,40,40,1,43.2,-16,-3
400 MAT PRINT T
410 PRINT
420 PRINT "DESTINATIONS"
430 PRINT "1. ANTARES"
440 PRINT "2. ANDROMEDA"
450 PRINT "3. TAU CETI"
460 PRINT "4. VEGA"
470 PRINT "5. SPICA"
480 PRINT "6. ARCTURUS"
490 PRINT "7. BETELGEUSE"
500 PRINT "8. ALPHA CENTAURI"
510 PRINT "9. ALPHA URSA MAJOR"
520 PRINT
530 PRINT
540 PRINT
550 PRINT
560 DISP "DESTINATION NUMBER";
570 INPUT N
580 REM
590 REM CONVERT DESTINATION COORDINATES
600 REM
610 J=T[N,3]
620 K=T[N,4]
630 U=T[N,1]
640 V=T[N,2]
650 C2=J+K/60
660 C=P-C2/X
670 E1=(U+V/60)*15
680 R=E1/X
690 S1=SINC
700 C1=COSC
710 PRINT N;U;V;J;K
720 PRINT E1;C1;C*X;S1;C1
730 PRINT
740 PRINT
750 REM
760 REM PLOT GRID
770 REM
780 READ X1,X2,X3,X4,X5,Y1,Y2,Y3,Y4,Y5
790 DATA 0,360,45,90,90,-90,90,45,90,90
800 GOSUB 1960
810 REM
820 REM ASSIGN FILES
830 REM
840 FOR N=5 TO 6
850 GOTO N OF 860,890,920,950,980,1010
860 FILES SD1
870 J=39
880 GOTO 1060
890 FILES SD2
900 J=57
910 GOTO 1060
920 FILES SD3
930 J=40
940 GOTO 1060
950 FILES SD4
960 J=60
970 GOTO 1060
980 FILES SD5
990 J=50
1000 GOTO 1060
1010 FILES SD6
1020 J=40
1030 REM
1040 REM RETRIEVE STAR DATA
1050 REM
1060 FOR I=1 TO J
1070 READ #1;B[I],C[I],D[I],E[I],F[I]
1080 PRINT B[I];C[I];D[I];E[I];F[I]
1090 NEXT I
1100 REM DETERMINATION OF TRIG FUNCTIONS OF DIHEDRAL AND CENTRAL ANGLES
1110 PRINT "DRA","O","DIHED ANG","CEN ANG"
1120 FOR I=1 TO J
1130 A[I]=SQR(1-B[I]^2)
1140 S=C[I]-R
1150 IF S>-PI THEN 1170
1160 S=S+2*PI
1170 IF S<PI THEN 1190
1180 S=S-2*PI
1190 O=SGNS
1200 U[I]=SIN(ABSS)
1210 V[I]=COSS
1220 REM DIHEDRAL ANGLE
1230 C3=U[I]/(S1*B[I]/A[I]-V[I]*C1)
1240 C4=ATNC3
1250 IF C4>0 THEN 1270
1260 C4=C4+PI
1270 G[I]=SINC4
1280 H[I]=COSC4
1290 REM CENTRAL ANGLE
1300 Y[I]=C1*B[I]+S1*A[I]*V[I]
1310 Z[I]=SQR(1-Y[I]^2)
1320 W=ATN(Z[I]/Y[I])
1330 IF W>0 THEN 1350
1340 W=W+PI
1350 IF I>10 THEN 1380
1360 PRINT S*X;O;C4*X;W*X;A[I]
1370 PRINT U[I];V[I];G[I];H[I];Z[I];Y[I]
1380 NEXT I
1390 REM
1400 REM DISTANCE LOOP
1410 REM
1420 FOR L=1 TO 4
1430 E=L[L]
1440 K=E^2
1450 PRINT
1460 PRINT "S/C DISTANCE NOW";E;"LIGHTYEARS"
1470 PRINT
1480 PRINT
1490 REM PRINT "S","D(I)","W1","W2","A*X","D","U1","V1"
1500 FOR I=1 TO J
1510 REM DETERMINATION OF DISTANCE FROM S/C TO OBSERVED STAR
1520 S=SQR(K+F[I]-2*E*D[I]*Y[I])
1530 W2=E[I]-7.566+5*LGTS
1540 IF W2>6 THEN 1870
1550 W1=(7-W2)^1.5/10
1560 REM DETERMINATION OF ANGLE FROM DESTINATION STAR TO OBSERVED STAR
1570 A=ATN(Z[I]/(E/D[I]-Y[I]))
1580 IF A>0 THEN 1600
1590 A=A+PI
1600 S2=SINA
1610 C2=COSA
1620 REM DETERMINATION OF DECLINATION OF OBSERVED STAR AT S/C
1630 C5=-C2*C1+S2*S1*H[I]
1640 S5=SQR(1-C5^2)
1650 U1=ATN(S5/C5)
1660 IF U1>0 THEN 1680
1670 U1=U1+PI
1680 U1=(P-U1)*X
1690 REM DETERMINATION OF RIGHT ASCENSION OF OBSERVED STAR AT S/C
1700 D=G[I]/(S1*C2/S2+C1*H[I])
1710 D=ATND
1720 IF D>0 THEN 1740
1730 D=D+PI
1740 V1=(R-PI-D*O)*X
1750 IF V1<360 THEN 1770
1760 V1=V1-360
1770 IF V1>0 THEN 1800
1780 V1=V1+360
1790 IF I>10 THEN 1810
1800 PRINT S;D[I];W1;W2;A*X;D*X;V1;U1
1810 PLOT V1,U1+W1
1820 PLOT V1+W1,U1
1830 PLOT V1,U1-W1
1840 PLOT V1-W1,U1
1850 PLOT V1,U1+W1
1860 PEN
1870 NEXT I
1880 PEN
1890 REM NEW COLOR
1900 STOP
1910 NEXT L
1920 PEN
1930 NEXT N
1940 GOTO 160
1950 END
1960 SCALE X1,X2,Y1,Y2
1970 XAXIS Y1,X3,X1,X2
1980 YAXIS X2,Y3,Y1,Y2
1990 XAXIS Y2,-X3,X2,X1
2000 YAXIS X1,-Y3,Y2,Y1
2010 FOR I=(Y1+Y4) TO (Y2-Y4) STEP Y4
2020 PLOT X1,I
2030 PLOT X2,I
2040 PEN
2050 NEXT I
2060 FOR I=(X1+X4) TO (X2-X4) STEP X4
2070 PLOT I,Y1
2080 PLOT I,Y2
2090 PEN
2100 NEXT I
2110 RETURN
```



```
10 REM RFD3 - BYBYBAYBEE - ADD STAR DATA TO OLD/NEW FILES -
20 REM DAT/TIM: 10AUG/12:48PM
30 DIM AS[100],BS[100],CS[100],DS[100],ES[100],FS[100]
40 DISP "NEW DATA FILE";
50 INPUT A
60 IF A=0 THEN 80
70 OPEN "SD6",3
80 X=180/PI
90 P=PI/2
100 REM NEW STAR ENTRIES
110 DISP "FIRST NUM OF NEW STAR ENTRIES";
120 INPUT I
130 I1=I
140 DISP "RA (HRS)";
150 INPUT U
160 DISP "RA (MIN)";
170 INPUT V
180 E1=(U+V/60)*15
190 C[I]=E1/X
200 DISP "DEC (DEG)";
210 INPUT J
220 DISP "DEC (MIN)";
230 INPUT K
240 C1=(J+K/60)
250 B[I]=COS(P-C1/X)
260 DISP "ABS MAG";
270 INPUT E[I]
280 DISP "DIST (LY)";
290 INPUT D[I]
300 PRINT I;U;V,J;K,E[I],D[I]
310 F[I]=D[I]^2
340 REM PRINT I;J;K;B;U;V;E1
350 I=I+1
360 GOTO 140
370 J=I-1
380 REM
390 REM FILE DATA
400 REM
410 FILES SD6
430 FOR I=I1 TO J
440 PRINT #1;B[I],C[I],D[I],E[I],F[I]
450 PRINT I;B[I];C[I];D[I];E[I];F[I]
460 NEXT I
470 PRINT
480 PRINT
490 REM CHECK: RETRIEVE DATA/CONVERT TO DEG/MIN/PRINT
500 READ #1,I1
510 FOR I=I1 TO J
520 READ #1;B[I],C[I],D[I],E[I],F[I]
530 PRINT I;B[I];C[I];D[I];E[I];F[I]
540 NEXT I
550 PRINT
560 PRINT
570 FOR I=I1 TO J
580 A[I]=SQR(1-B[I]^2)
590 B=90-ATN(A[I]/B[I])*X
600 IF B <= 90 THEN 620
610 B=B-180
620 A=INTB
630 B=(B-A)*60
640 C=C[I]*X/15
650 F=INTC
660 C=(C-F)*60
670 PRINT I;F;C;A;B;E[I];D[I]
680 NEXT I
690 END
```

```
10 REM RFD5 - BYBYBAYBEE:DATA CHECKOUT: RETRIEVE/CONVERT/PLOT NORMAL STAR MAP-
20 REM DAT/TIM: 18AUG/ 7:36AM
30 REM  DATA CHECKOUT: ASN FILES; GET DATA; CNV TO DEC/RA; CNV TO REL MAG; PLOT
40 DIM AS[100],BS[100],CS[100],DS[100],ES[100],FS[100]
50 X=180/PI
60 K=0
70 REM
80 REM ASSIGN FILES
90 REM
100 FOR N=1 TO 6
110 GOTO N OF 120,150,180,210,240,270
120 J=39
130 FILES SD1
140 GOTO 320
150 J=57
160 FILES SD2
170 GOTO 320
180 J=40
190 FILES SD3
200 GOTO 320
210 J=60
220 FILES SD4
230 GOTO 320
240 J=50
250 FILES SD5
260 GOTO 320
270 J=40
280 FILES SD6
290 REM
300 REM RETRIEVE AND PRINT DATA
310 REM
320 FOR I=1 TO J
330 READ #1;B[I],C[I],D[I],E[I],F[I]
340 PRINT I;B[I];C[I];D[I];E[I];F[I]
350 NEXT I
360 PRINT
370 REM
380 REM PLOT GRID
390 REM
400 K=K+1
410 IF K>1 THEN 480
420 READ X1,X2,X3,X4,X5,Y1,Y2,Y3,Y4,Y5
430 DATA 0,360,45,90,90,-90,90,45,90,90
440 GOSUB 740
450 REM
460 REM CONVERT/PLOT/PRINT
470 REM
480 FOR I=1 TO J
490 A=SQR(1-B[I]^2)
500 B=90-ATN(A/B[I])*X
510 IF B <= 90 THEN 530
520 B=B-180
530 A=INTB
540 B1=INT((B-A)*60)
550 C=C[I]*X
560 F=INT(C/15)
570 C1=INT((C/15-F)*60)
580 PRINT I;F;C1;A;B1;D[I];E[I]
590 S=D[I]
600 W2=E[I]-7.566+5*LGTS
610 IF W2>6 THEN 700
620 W1=(7-W2)^1.5/10
630 REM PRINT S,E[I],W2,W1
640 PLOT C,B+W1
650 PLOT C+W1,B
660 PLOT C,B-W1
670 PLOT C-W1,B
680 PLOT C,B+W1
690 PEN
700 NEXT I
710 PEN
720 NEXT N
730 END
740 SCALE X1,X2,Y1,Y2
750 XAXIS Y1,X3,X1,X2
760 YAXIS X2,Y3,Y1,Y2
770 XAXIS Y2,-X3,X2,X1
780 YAXIS X1,-Y3,Y2,Y1
790 FOR I=(Y1+Y4) TO (Y2-Y4) STEP Y4
800 PLOT X1,I
810 PLOT X2,I
820 PEN
830 NEXT I
840 FOR I=(X1+X4) TO (X2-X4) STEP X4
850 PLOT I,Y1
860 PLOT I,Y2
870 PEN
880 NEXT I
890 RETURN
```

PROGRAMS
PROBLEMS & SOLUTIONS

BY JIM CONLAN

The definition of a prime number goes back at least 2000 years to good old Euclid. Euclid was aware that it is possible to arrange six little squares into a rectangle having two rows and three columns. He also knew there is no way to arrange seven little squares into a rectangle, except by putting them in one long row. This observation might seem trivial until you try a slightly larger number. Can you tell quickly whether 6889 little squares can be arranged in rows and columns to form a big rectangle? One long row always works, so that is not the answer.

Today we tend to think more algebraically and say that a number is prime if it has no divisors except the number itself and 1. A number is composite if it has divisors other than 1 and the number itself. The numbers 2, 3, 5, 7, 11, and 13 are prime. The numbers 4, 6, 8, 9, 10, 12 and 6889 are composite (6889 = 83 * 83).

Tables showing the factors of numbers have been laboriously computed over the centuries by industrious lovers of numbers. A modern table of prime factors of numbers is published in the *Handbook of Mathematical Functions* edited by Abrahamowitz and Stegun. This table gives the factorizations of all numbers up to 9999. The *Handbook* also contains a list of all prime numbers up to 99991. This list was taken from an even larger list of primes up to 10,006,721 published by D. N. Lehmer in 1914. Lehmer is famous even today for the ingenious method he devised to determine the primality of large numbers. His list was the result of years of arduous thought and dedicated effort.

Research into methods of finding prime factors of large numbers received added practical motivation in 1977 when Rivest, Shamir, and Adleman, working with an idea first proposed by Diffie and Hellman, showed how to develop secret codes which could not be broken unless one could find a way to factor very large numbers. Since no one has ever found any fast way to do this, it is believed that these codes are, for all practical purposes, unbreakable.

A more common use of prime numbers and factorizations of numbers into primes is to find the sum or product of two fractions. Much of the difficulty of working with fractions arises because of the difficulty of finding the prime factors of the numbers involved. Try adding 5/2077 to 13/2479 and reducing the result to see how difficult a fraction problem can be.

Problem #5 asked for a subroutine to determine whether a positive integer NPT is a prime number.

> If NPT is not a prime number, set PPT=0 and set FPT equal to a factor of NPT other than 1 or NPT (1<FPT<NPT), then return.
>
> If NPT is a prime number, set PPT=1 and set FPT=1, then return.

10,006,721

The subroutine will have one input (NPT) and two outputs (PPT and FPT). There were two small constraints:

The subroutine should be fast.

Don't use arrays. Especially, don't store a bunch of primes.

The most straight-forward routine just starts dividing until a divisor is found. How does one determine whether one number divides another exactly? Suppose one wishes to determine whether 43 is exactly divisible by 17. The quotient 43/17 = 2.5294. The decimal part indicates that 17 is not a divisor of 43. On the other hand, since 51/17 = 3. we see that 51 is divisible by 17. The integer part of the quotient is identical to the quotient itself. This can be expressed in BASIC by

INT(51/17)=51/17

Here is a simple program which uses this test. This program divides the number NPT by all the numbers IPT from 2 to NPT-1. NPT always divides NPT so we don't need to try that.

```
510 FOR IPT TO NPT-1
515 REM CHECK IF IPT DIVIDES NPT
520 IF INT(NPT/IPT)=NPT/IPT THEN
    GOTO 580
530 NEXT IPT
540 REM NO DIVISORS FOUND
550 PPT=1
560 FPT=1
570 RETURN
580 REM IPT IS A DIVISOR OF NPT
590 PPT=0
600 FPT=IPT
610 RETURN
```

This routine can be improved dramatically by a moment's thought. Consider how the previous routine would deal with the prime number 101. The previous routine would divide 101 by 2, 3, 4, 5, and so on, up to 99. This is exceedingly wasteful. There is no need to divide by any number after 10. If no number from 2 to 10 divides 101, then no number larger than 10 can be a divisor of 101. Why? How many times does 11 divide into 101? Certainly less than 10 times. If 11 divides 101 exactly, then so does some small number. If none of the numbers 10 or less divide 101 exactly, then neither can the numbers 11 or larger. If NPT is the product of two numbers A and B, then one of the numbers must be no larger than the square root of NPT. The trial divisors IPT never need to be larger than SQR(NPT). Using this observation will allow our program to run thousands of times faster on large numbers. The only change we need to make is to replace line 510 with

510 FOR IPT=2 TO SQR(NPT)

Another observation yields more savings: if 2 won't divide NPT, then neither will 4, 6 or any other even number. We only need to try odd divisors after we have tried 2. Here is a routine which uses this observation.

```
500 REM CHECK IF 2 DIVIDES NPT
502 IPT=2
504 IF INT(NPT/IPT)=NPT/IPT
    THEN 580
508 REM CHECK ODD DIVISORS IPT
510 FOR IPT=3 TO SQR(NPT) STEP 2
520 IF INT(NPT/IPT)=NPT/IPT
    THEN 580
530 NEXT IPT
540 REM NO DIVISORS FOUND
550 PPT=1
560 FPT=1
570 RETURN
580 REM IPT IS A DIVISOR OF NPT
590 PPT=0
600 FPT = IPT
610 RETURN
```

The previous routine found that 10,006,721 was a prime number in about 47 seconds on an Atari 800 computer. Remember that this was the largest prime in D. H. Lehmer's 1914 table of primes.

Eratosthenes was the ablest scientist of antiquity. He was the librarian of the Greek library at Alexandria in Egypt around 200 AD. He was nicknamed Beta (the second letter of the Greek alphabet) because he was the second best wrestler, the second best poet and the second best mathematician in Alexandria. He was the first person to measure the circumference of the earth.

We remember Eratosthenes today because of a simple and ingenious method which he developed for finding prime numbers. We will use his method to find the primes up to 4000.

Consider the sequence of 4000 boxes which are partially pictured in Figure 1. Each box contains the symbol "1."

```
|1|1|1|1|1|1|1|1|1|1|1|1|1|1|1|1|1|1|. . . .
 1 2 3 4 5 6 7 8 9 . . . . . . . . . . . . . .
```

Figure 1.

The number 2 is prime since it has no divisors except itself and 1. If we mark every second box following 2 with the symbol "2" then we will have marked all the multiples of 2 with a "2."

```
|1|1|1|2|1|2|1|2|1|2|1|2|1|2|1|2|1|2|. . . . .
 1 2 3 4 5 6 7 8 9 . . . . . . . . . . . . . .
```

The next number after 2 which has not been marked is 3. Since 3 is not marked out, it is not a multiple of 2. Thus 3 has no divisors smaller than itself and is therefore, prime.

If we mark every third number following 3 with the symbol "3" then we will mark all those numbers which are multiples of 3.

```
|1|1|1|2|1|3|1|2|3|2|1|3| . . . . . . . . . . .
 1 2 3 4 5 6 7 8 9 . . . . . . . . . . . . . .
```

The next number after 3 which is not marked is 5. Since 5 is not marked, it is not a multiple of 2 or 3. Thus, it has no prime divisors smaller than itself and is, therefore, prime.

If we mark every fifth number following 5 with the symbol "5" then we will mark all those numbers which are multiples of 5.

```
|1|1|1|2|1|3|1|2|3|5|1|3| . . . . . . . . . . .
 1 2 3 4 5 6 7 8 9 . . . . . . . . . . . . . .
```

You might find it instructive to do the next case after 5. Consider this question: Where is the first number you will need to mark? It is not really 14 or 21 since they have already been marked.

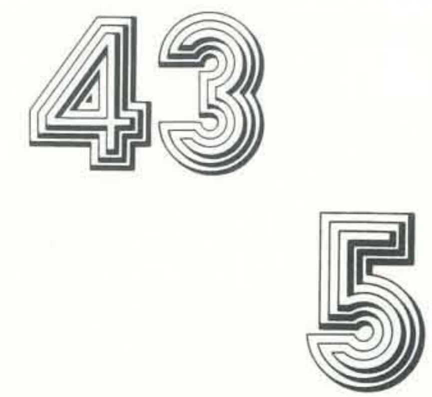
You may notice that the next number that needs to be marked is 7*7=49. All earlier multiples of 7 have already been crossed out. Where does one start crossing out multiples of 11? Where does one start marking multiples of 67? (67*67=4489, thus all the multiples of 67 which are less than 4000 will have been previously marked.)

The following program utilizes this sieve method first used by Eratosthenes.

This routine creates a string P$ consisting of 4000 characters. If number N is prime, then the ASCII number of the character is 1. If N is composite, then the ASCII number of the character is a prime divisor of N.

The number 4000 was chosen for illustrative purposes only. (If your computer allows only short strings, you may need to replace 4000 with 255.) A computer with sixty-four thousand 8-bit bytes of available memory could handle all the primes to 64000.

The computational time varies in direct proportion to the number of primes computed. If the first 4000 primes take 1 minute on a machine, the 64000 primes will take 16 minutes.



```
2 REM : COMPUTE STRING P$ OF
  DIVISORS
10 REM : SET UP L BOXES
15 L=4000
20 DIM P$(L)
30 REM : P=2 IS THE INITIAL PRIME
40 P=2
50 REM : PUT CHARACTER NUMBER
   1 IN ALL THE BOXES
60 FOR I=1 TO L
70 P$(I,I)=CHR$(1)
80 NEXT I
90 REM : CHECK P TO SEE IF DONE
   MARKING BOXES
100 IF P*P>L THEN 150
110 REM : MARK ALL MULTIPLES
    OF P WITH CHARACTER P
120 FOR K=P*P TO L STEP P
130 P$(K,K)=CHR$(P)
140 NEXT K
150 REM : CHECK THE NEXT BOX.
    ARE WE DONE?
160 P=P+1:IF P>L THEN 210
170 REM : HAVE WE FOUND THE
    NEXT PRIME YET?
180 IF P$(P,P)=CHR$(1) THEN 100
190 REM : IF BOX P DOESN'T
    CORRESPOND TO A PRIME
    THEN GO TO NEXT
200 GOTO 150
210 PRINT "FACTOR TABLE IS
    COMPLETE"
220 PRINT "WHAT NUMBER DO YOU
    WISH TO FACTOR?"
230 INPUT N
240 PRINT ASC(P$(N))
250 GOTO 220
```

Here is a condensed version of the previous program. This eight-line program computed a prime factor for each of the numbers from 1 to 4000 in 55 seconds on an Atari 800 computer.

```
10 L=4000:P=2:DIM P$(L)
20 FOR I=1 TO L:P$(I,I)=CHR$(1):
   NEXT I
30 IF P*P>L THEN 50
40 FOR K=P*P TO L STEP P:P$(K,K)=
   CHR$(P):NEXT K
50 P=P+1:IF P>L THEN 70:IF P$(P,P)=
   CHR$(1) THEN 30:GOTO 50
60 ? "NUMBER TO BE FACTORED?"
70 INPUT N
80 PRINT ASC(P$(N,N)):GOTO 70
```

The previous routine does not satisfy the restrictions of problem #5 which specifically said we were not to use arrays. This routine uses a dimensioned string array. This routine has one advantage: by saving the results in a string, the factorization of numbers can be done very quickly, without any noticeable waiting. This is appropriate for a program which works with relatively small (64000 or less) numbers. A program to add and multiply fractions would need to work "instantaneously" without any frustrating waits. The sieve method would be an appropriate subroutine.

We can use a variant of the sieve method to speed up the previous division routines. Observe the pattern in the sieve after multiples of 2 and 3 have been marked with *.

```
|1|*|*|*|1|*|1|*|*|*|1|*|1|*|*|*|1|*|
|1 2 3 4 5 6|7 8 9 1 1 1|1 1 1 1 1 1|
|           |      0 1 2|3 4 5 6 7 8|
```

Notice that the pattern of 1's and *'s is periodic. The pattern repeats with a period of 6. The possible primes are separated by jumps of three *'s then two *'s then three *'s then two *'s and so on and on. The following subroutine uses these observations to speed up the testing process. The routine first tests for divisibility by 2 and 3. Next the routine jumps 4 places (over three *'s) to 5. Next the routine jumps two places (over one *) to 7 and so on.

```
500 REM SUBROUTINE TO FIND
    PRIME DIVISORS OF NPT
505 REM ************************
510 S=SQR(NPT)
515 REM ************************
520 REM CHECK FOR DIVISIBILITY
    BY 2,3
530 I=2:GOSUB 660
540 I=3:GOSUB 660
545 REM ************************
550 I=1
555 REM ************************
560 REM INCREMENT DIVISOR I
    AND GO CHECK DIVISIBILITY
570 I=I+4:GOSUB 660
580 I=I+2:GOSUB 660
590 GOTO 570
655 REM ************************
660 REM CHECK IF DONE
670 IF I>S THEN 720
675 REM ************************
680 REM CHECK DIVISIBILITY
690 IF NPT/I=INT(NPT/I) THEN 730
700 RETURN
705 REM ************************
710 REM NO DIVISORS. NPT IS PRIME
720 PPT=1:FPT=1:POP :RETURN
722 REM ************************
725 REM I IS A DIVISOR OF NPT
730 PPT=0:FPT=I: POP :RETURN
```

This routine took 39 seconds to find that 10,006,721 is prime. The command POP on lines 720 and 730 is used to POP the subroutine stack on the Atari 800. The RETURN is from subroutine 500.

The method of the last routine can be improved still further. Consider the pattern of the sieve after all multiples of 2, 3 and 5 have been marked with an *.

```
|1|*|*|*|*|*|1|*|*|*|1|*|1|*|*|*|1|*
|1|*|*|*|1|*|*|*|*|*|1|*|
```

This pattern repeats again starting after 30. The jumps to the next possible primes are 6, 4, 2, 4, 2, 4, 6 and 2. A slight modification of the previous subroutine makes use of this observation. Replace and insert the following lines:

```
545 I=5:GOSUB 660
570 I=I+6:GOSUB 660
580 I=I+4:GOSUB 660
590 I=I+2:GOSUB 660
600 I=I+4:GOSUB 660
610 I=I+2:GOSUB 660
620 I=I+4:GOSUB 660
630 I=I+6:GOSUB 660
640 I=I+2:GOSUB 660
650 GOTO 570
```

This last routine determined that 10,006,721 was prime in 33 seconds.

All the previous routines started trying small divisors first. If the number has only large divisors, then this method might be painfully slow. It is a remarkable fact, discovered in the early 17th century by Pierre Fermat, that there is a method which finds large factors first. Fermat's factorization method depends on a simple fact of algebra which one can easily check.

N=U*V if an only if

N=X↑2−Y↑2 where X=1/2*(U+V) and

Y=1/2*(U−V)

In order to see if N=U*V for some numbers U and V we need only check if N=X↑2−Y↑2 for some numbers X and Y. This turns out to be relatively easy when U and V are about the same size (both relatively large factors). Put in a slightly different algebraic form, we need to determine whether there is an integer X equal to SQR(N+Y↑2). The following routine uses this method of Fermat.

```
10 REM FERMAT'S FACTORIZATION
20 INPUT N
30 FOR Y=0 TO N/2
40 X=SQR(N+Y*Y)
50 IF X=INT(X) THEN 80
60 NEXT Y
70 PRINT N;" IS PRIME":GOTO 20
80 U=X+Y
90 V= X-Y
100 PRINT N;"=";U;"*";V:GOTO 20
```

The previous routine factored 1726547 = 1321*1307 in less than 1.5 seconds (on an Atari 800). Division routines would have taken approximately four times as long. ■

Mark of Breed-ing

by Chuck Upmann

Illustration by Barbara Ruzgerian.

"Interesting, very interesting," was Commander Steelorb's comment as he and the ever-faithful Ensign Landry were shown about the computer breeding ranch. He paused for a moment to light his battered old pipe; soon pungent clouds of green Vegan swampweed smoke wreathed his impressive mein.

"Thank you," replied a proud but harried Pacot Lumesh. "Glad you like my little endeavor. Always a pleasure to host a visit from the Space Survey Corps." He wiped sweat from his head with a soft square of red plastic. Pacot Lumesh pointed to the awkward creatures on the other side of a near-by fence. "Newborn computer colts."

"Hmmm," replied the Commander. Puff, puff on the pipe. "Terribly fascinating, the raising of organic computers." He watched the long-legged youngsters cavort around the grassy field. The colts resembled nothing so much as metallic cucumbers on long, spindly legs. Rudimentary fins were beginning to sprout from their rounded posteriors.

"Not an easy job, not atall," Pacot Lumesh was quick to affirm. They moved to the next pasture. Here the fences were much higher and supplemented by force fields. The organic computers inside were larger, obviously adult. They had no legs, only the large fins on the rear portion of their bodies. They floated slightly above the ground level, dipping now and then to pluck a mouthful of grass. The fins moved fractionally, controlling their direction of travel. The edges looked very sharp to Commander Steelorb.

"They eat grass?" Ensign Landry asked unnecessarily.

"Efficiency in the extreme," answered the Commander. "They convert organic stuff to electrical energy."

"We breed the best organic computers in all of the Gordon Cluster," Pacot Lumesh said in a most prideful manner. Then he seemed to remember that he spoke to the august personnage of the Space Survey Corps' most reknowned troubleshooting officer. He gulped nervously. "All except for a few," he finished weakly.

Steelorb nodded. "So I heard."

"Yeah, some of them didn't mutate properly under the modification beam," Ensign Landry said smugly.

"Shut up, Landry," the Commander said abstractly as he relit his pipe. He gestured toward yet another field. "What're those chaps doing?" Lumesh turned and saw several of his employees busily engaged in filing the fins of several adult organic computers.

"They're dulling the edges so the critters will mate," he explained. "Won't go near each other with sharp fins, y'know."

"Don't blame them," said Ensign Landry.

"Umphf," said Commander Steelorb, with several puffs on his odiferous pipe. "Let me be certain I understand this procedure. You radiate the computers and allow them to mate, dulling the fins so that they will. This determines the type of organic computer produced."

"Precisely," Pacot Lumesh answered.

"But it hasn't always given you the correct result. Has it now?" said Ensign Landry.

Pacot Lumesh became slightly angry, but mindful of who he was with, remained respectful to Commander Steelorb. They both ignored young Landry. "That was indeed a severe problem for a time," he said. "But we have isolated the defective breeding stock and have only to brand them so that no mix-ups will occur."

"Excellent," replied Steelorb. "So our professional help is unneeded. Except, perhaps, for wording the mark to be applied to the bad breeders."

"That has been a problem," Pacot Lumesh answered, nodding his head.

"Well. With the data you gave us, the wording is obvious in the extreme."

Lumesh waited expectantly. Ensign Landry, well acquainted with the Commander's excesses, was quietly sobbing into his hands.

Commander Steelorb nodded to himself. "Yes. Stamp them: 'Do not foal, fin dull, or mutate.' " ■

by Curtis Cooper

The Fifteen Puzzle

The Fifteen Puzzle, a creation of puzzle wizard Sam Loyd, is a mathematical problem which can be simulated on a microcomputer. This game consists of 15 numbered squares and a space in a 4 x 4 tray. The puzzle starts with some permutation of the numbered tiles. The object of the Fifteen Puzzle is to rearrange the tiles by sliding them about the tray, until the numbered squares are in serial order. For example, the initial configuration of the Fifteen Puzzle may look like Figure 1 and the final, sought-after configuration is Figure 2.

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

Figure 1

15	7	4	14
9	3	11	
13	8	10	6
5	2	12	1

Figure 2

Program Notes

The North Star BASIC program will let you enjoy this game and help improve your skill at solving it. It uses two arrays, an array ("A") dimensioned to 16 and an array T dimensioned to 16 x 5. The first array consists of the numbers 1 through 15 and the number 0, which represents the space. We can think of it as the puzzle board. With respect to the puzzle board in Figure 1, the array would appear as in Figure 3.

15	7	4	14	9	3	11	0	13	8	10	6	5	2	12	1
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Figure 3 – Array "A".

Array T denotes the neighbors of a square on the puzzle board. For example, square 2 has 3 neighbors (square 1, square 3, and square 6) while square 7 has 4 neighbors (square 3, square 6, square 8, and square 11). T(i,1) represents the number of neighbors of square i, and T(i,2) through T(i,5) contains the list of those neighbors, filled-out with zeros if necessary. Thus, with respect to the finished puzzle (Figure 2), the two examples above would appear as

T(2,1) = 2	T(7,1) = 4
T(2,2) = 1	T(7,2) = 3
T(2,3) = 3	T(7,3) = 6
T(2,4) = 6	T(7,4) = 8
T(2,5) = 0	T(7,5) = 11.

The program begins by explaining the Fifteen Puzzle. It asks for the difficulty level you wish to attempt and then randomly generates your scrambled puzzle. The program next prints out the arrangement of the numbers on the board and asks you to indicate the number of the tile you wish to move to the blank square. It moves your indicated tile to the blank space and checks to see if the tiles are in serial order. If they are, the number of moves you took to arrange the tiles in order is printed and you are asked if you want to try another puzzle. If they are not in serial order, you are asked to give another move, etc.

Figure 4 is a sample run of the program. The program listing begins on page 35. ■

References

1. Gardner, Martin. *Scientific American*, Vol. 197 (Aug. 1957), 120.
2. Gardner, Martin. *Scientific American*, Vol. 210 (Feb. 1964), 122.
3. Liebeck, Hans. "Some Generalizations of the 14-15 Puzzle," *Mathematics Magazine*, Vol. 44 (1971), 185-189.
4. Spitznagel, Edward L., Jr. "A New Look at the Fifteen Puzzle," *Mathematics Magazine*, Vol. 40 (1967) 171-174.

Figure 4.

```
SAMPLE RUN

THIS PROGRAM SIMULATES THE
FAMOUS 'FIFTEEN PUZZLE'. THE
OBJECT OF THE GAME IS TO START
WITH A SCRAMBLED 4X4 ARRAY OF
15 NUMBERED TILES AND A SPACE
AND ARRANGE THE TILES IN
ORDER FROM 1 TO 15 WITH THE
SPACE IN THE LOWER RIGHT
CORNER.

WHAT LEVEL OF DIFFICULTY DO
YOU WISH? (1,2,3,4) (4 IS HARDEST)
? 1

 6     1     7     2
10     3     8    12
 0     5    14     4
13     9    11    15

INDICATE THE NUMBER OF THE
TILE YOU WISH TO MOVE TO THE
BLANK SQUARE ? 10

 6     1     7     2
 0     3     8    12
10     5    14     4
13     9    11    15

INDICATE THE NUMBER OF THE
TILE YOU WISH TO MOVE TO THE
BLANK SQUARE ? 6

 0     1     7     2
 6     3     8    12
10     5    14     4
13     9    11    15

INDICATE THE NUMBER OF THE
TILE YOU WISH TO MOVE TO THE
BALNK SQUARE ? 1

.
.
.

 1     2     3     4
 5     6     7     8
 9    10    11    12
13    14     0    15

INDICATE THE NUMBER OF THE
TILE YOU WISH TO MOVE TO THE
BLANK SQUARE ? 15

 1     2     3     4
 5     6     7     8
 9    10    11    12
13    14    15     0

CONGRATULATIONS! ! ! YOU HAVE
FINALLY ARRANGED THE TILES
IN THE CORRECT ORDER. YOU
TOOK 156 MOVES.

DO YOU WISH TO TRY ANOTHER
POSSIBILITY ? (YES/NO) ? NO

GOODBYE. . .THANKS FOR PLAYING.
```

```
10 REM ************ THE FIFTEEN PUZZLE ************
20 REM ************            BY            ************
30 REM ************    CURTIS N. COOPER    ************
40 REM ************  SEPTEMBER 15, 1980  ************
50 DIM T(16,5),A(16)
60 FOR I = 1 TO 16
70 READ T(I,1),T(I,2),T(I,3),T(I,4),T(I,5)
80 NEXT I
90 PRINT "THIS PROGRAM SIMULATES THE FAMOUS"
100 PRINT " 'FIFTEEN PUZZLE'. THE OBJECT OF"
110 PRINT "THE GAME IS TO START WITH A"
120 PRINT "SCRAMBLED 4X4 ARRAY OF 15 NUMBERED"
130 PRINT "TILES AND A SPACE AND ARRANGE THE"
140 PRINT "TILES IN ORDER FROM 1 TO 15 WITH THE"
150 PRINT "SPACE IN THE LOWER RIGHT CORNER."
160 PRINT
170 J1=RND(-1)
180 PRINT
190 INPUT "WHAT LEVEL OF DIFFICULTY DO YOU WISH?
    (1, 2, 3, 4) (4 IS HARDEST) ",C1
200 PRINT\PRINT\PRINT\PRINT\PRINT\PRINT
210 REM THIS SECTION RANDOMLY GENERATES A FIFTEEN
    PUZZLE OF ***************
220 REM OF A PREDETERMINED DIFFICULTY ****************
230 PRINT
240 C=0
250 FOR I = 1 TO 15
260 A(I) = I
270 NEXT I
280 A(16) = 0
290 K1 = 16
300 K0 = 0
310 J1 = RND(J1/100)
320 FOR I = 1 to C1*50
330 J1 = RND(J1)
340 L = INT(T(K1,1)*J1+1)
350 K2 = T(K1,L+1)
360 IF K2=K0 THEN 330
370 K = A(K1)
380 A(K1) = A(K2)
390 A(K2) = K
400 K0 = K1
410 K1 = K2
420 NEXT I
430 REM  **********************************************
440 PRINT\PRINT\PRINT\PRINT\PRINT
450 GOSUB 990
460 PRINT\PRINT\PRINT\PRINT\PRINT
470 PRINT "INDICATE THE NUMBER OF THE TILE"
480 PRINT "YOU WISH TO MOVE TO THE BLANK SQUARE"
490 INPUT M
500 C = C + 1
510 PRINT CHR$(12)
520 FOR J = 1 TO 16
530 IF A(J) = M THEN EXIT 590
540 NEXT J
550 PRINT "INVALID INPUT, TRY AGAIN"
560 GOTO 440
```

continued on page 36.

The Next Whole Earth Catalog
Edited by Stewart Brand
Published by Random House
608 pages, $12.50 paperbound
Reviewed by Julie Anton

First it was *The Whole Earth Catalog*, homely Bible of a generation who had dropped out and turned on in the 1960's, turned off a few years later, and split for the hills in search of something safe and homespun.

Whole Earth chronicler Stewart Brand watched his brainchild move from commune floors to chrome suburban coffee tables, hit the international best-sellers list, and win a controversial National Book Award. Troubled by fame, Brand officially called it quits with *The Last Whole Earth Catalog* in 1971. He managed one more *Epilogue* in 1974, but the Whole Earth flame was definitely out. He observed that "you cannot both be and defend a mountain range."

Now we have *The Next Whole Earth Catalog*. This is a 608-page distillation of the best elements of the earlier catalogs, with a vivid transformation for the 1980's, granting snap and modernity to the sense of practical pioneering and self-reliance which made its predecessors so important and unique.

"The image of me and mah woman and mah kids and mah dog and the chickens and the ducks and cows and the woods is a self-destructive fantasy after a while," Stewart Brand says, "because it eliminates a lot of quite wholesome dependencies from one's thinking." His latest catalog is not a Book of Exodus for a lost generation. We are not looking to escape from the 1980's. We are involved, and we like it that way.

Stewart Brand describes the purpose of his updated Whole Earth Catalog this way: "We are as gods and might as well get good at it. So far remotely done power and glory – as via government, big business, formal education, church – has succeeded to the point where gross defects obscure actual gains. In response to this dilemma and to these gains a realm of intimate, personal power is developing – the power of individuals to conduct their own education, find their own inspiration, shape their own environment, and share the adventure with whoever is interested. Tools that aid this process are sought and promoted by *The Next Whole Earth Catalog*."

There could be no greater symbol of this new Whole Earth philosophy than the computer, a tool which Stewart Brand expects will change our lives more

(continued on page 51)

The Fifteen Puzzle

Continued from page 35

```
570 REM THIS SECTION INTERCHANGES THE BLANK SPACE *******
580 REM AND THE NUMBERED TILE YOU MOVED ****************
590 I = J
600 K = T(I,1)
610 FOR J = 1 TO K
620 L = T(I,J+1)
630 IF A(L) = 0 THEN EXIT 660
640 NEXT J
650 GOTO 550
660 K = T(I,J+1)
670 J = A(I)
680 A(I) = A(K)
690 A(K) = J
700 GOSUB 990
710 FOR J = 1 TO 15
720 IF A(J) <> J THEN EXIT 460
730 NEXT J
740 PRINT " CONGRATULATIONS! ! ! YOU HAVE FINALLY
    ARRANGED THE TILES "
750 PRINT " IN THE CORRECT ORDER. YOU TOOK ",C," MOVES."
760 PRINT
770 INPUT " DO YOU WISH TO TRY ANOTHER POSSIBILITY ?
    (YES/NO) ",Y$
780 IF Y$ = "YES" THEN 180
790 PRINT " GOODBYE..........THANKS FOR PLAYING." END
800 REM THIS SECTION DEFINES THE NEIGHBORS OF EACH ******
810 REM SQUARE ON THE FIFTEEN PUZZLE BOARD *************
820 DATA 2,2,5,0,0
830 DATA 3,1,3,6,0
840 DATA 3,2,4,7,0
850 DATA 2,3,8,0,0
860 DATA 3,1,6,9,0
870 DATA 4,2,5,7,10
880 DATA 4,3,6,8,11
890 DATA 3,4,7,12,0
900 DATA 3,5,10,13,0
910 DATA 4,6,9,11,14
920 DATA 4,7,10,12,15
930 DATA 3,8,11,16,0
940 DATA 2,9,14,0,0
950 DATA 3,10,13,15,0
960 DATA 3,11,14,16,0
970 DATA 2,12,15,0,0
980 REM THIS SUBROUTINE PRINTS THE FIFTEEN PUZZLE BOARD *
990 PRINT %10I,A(1),%5I,A(2),%5I,A(3),%5I,A(4)
1000 PRINT
1010 PRINT %10I,A(5),%5I,A(6),%5I,A(7),%5I,A(8)
1020 PRINT
1030 PRINT %10I,A(9),%5I,A(10),%5I,A(11),%5I,A(12)
1040 PRINT
1050 PRINT %10I,A(13),%5I,A(14),%5I,A(15),%5I,A(16)
1060 PRINT
1070 RETURN
```


ComputerTown Visits with Senior Citizens

by Patricia Smith

Bill Scarvie

Often a real gap in understanding exists between people who work with computers and people who don't, the latter often viewing the former as hopelessly lost in outer space. ComputerTown, USA!, the grass roots community organization established to bring computer literacy to the general community, recently attempted to bridge this gap by showing a group of senior citizens how computers work. Geri Foley, Program Director of the Little House multi-service center for seniors in Menlo Park, California, explained she was excited when the ComputerTown, USA! coordinator contacted her about setting up the demonstration. Such a program, she felt, would not only help increase awareness about computers, but would also help bridge the generation gap.

Foley said that when she first told some of the Little House members that volunteers were coming with microcomputers, a lot of people were afraid. But she said the program, which took place December 11, 1980, in the cafeteria after lunch, did attract a lot of attention and some fifty people remained for the activities.

During the program, Little House members were able by playing games to get a feel for how computers work. Charles Lewis commented he played one game, but mainly watched. "I was mostly interested in how they program," he said.

Lewis explained he was particularly interested in the demonstration because one of his sons is studying computer science and another one uses computers in his work, and he wanted some information about what his children are doing.

Many people in the audience commented they enjoyed playing the games. Eleanor Weber, past president of Little House, and Doris Orner both enjoyed playing Black Jack. Orner said, "I won every time!"

Lorene Hoffman, a Little House volunteer and associate editor of the Little House News, said, "I just loved it. I've watched star games over kids' shoulders but never played before. I'd love to have a computer, and I'd probably give up bridge unless it's programmed for bridge," she said.

A few people sat in the back of the cafeteria and did not try to use or watch others use the demonstration microcomputers. One woman stated, "I live alone and wouldn't play games there."

But most people attending seemed to understand that not only could games be played on the computer, but more importantly, the computers could be programmed to assist with everyday calculations and tasks. At the end of the session, one man was interested in learning more about how microcomputers can assist with financial calculations. Several people who are writers were interested in learning more about word processing. One woman saw the computer as a potential memory aid; she could store shopping lists and other information she might forget.

In addition to bridging the generation gap between computer users and non-users, Foley said she is interested in ways computers can help organizations like Little House provide better service for their members. She explained that Little House membership is open to any adult over fifty years of age and a variety of educational programs and social services are offered to members. There are now around 3,000 members and an additional 7,000 people have participated in Little House activities. Foley said that if they had information on file in a computer about the interests of members and other potential members who stopped by the center at one time or another, they could plan to better meet the needs of the community they serve. For example, "We could pull out and see that someone joined in October but hasn't been here after that, and find out why," she said.

She added that with computerized information they could localize where their members live and then arrange for car pool services for people who call and say, "I'd really like to come, but I don't have a ride."

Foley said the ComputerTown, USA! program was a success and she hopes to have the group back again. ■

Electric Phone Book

A computerized bulletin board works just like an ordinary bulletin board system except that instead of paper and thumbtacks it uses a terminal, a computer, and the dial-up telephone network.

The list below was developed from several sources including the Peripheral People in Mercer Island, Washington and the People's Message System in Santee, California. It is being maintained by People's Computer Company's PCNET project, our effort to bring computers and telecommunications into the hands of everyone. While this is the most complete listing we have as of this writing, we would appreciate additions and corrections. Send them to PCNET, PCC, P.O. Box E, Menlo Park, CA 94025.

All the bulletin board systems listed here can be accessed by telephone using a 300-baud ASCII terminal and a Bell 103 modem. Most use carriage-return as a speed recognition character, after which they are self-teaching. All are free to anyone who calls, unlike the Arpanet, which is restricted, and The Source and MicroNet, which cost money. The list has been sorted by area code; consult your local telephone directory for geographical correspondence. ■

(201) 283-2724	**(214) 288-4859**	**(414) 241-8364**	**(713) 233-7943**
(201) 457-0893	(214) 634-2668	(414) 282-8118	(713) 693-8080
(201) 688-7117	**(214) 634-2775**	**(415) 348-2139**	**(713) 977-7019**
(201) 753-1225	(214) 641-8759	(415) 348-2396	(714) 449-5689
(201) 753-8152	**(216) 644-1965**	**(415) 493-7691**	**(714) 463-0461**
(201) 835-7228	(216) 754-7855	(415) 527-0400	(714) 495-6458
(201) 843-4563	**(301) 344-9156**	**(415) 647-9524**	**(714) 526-3687**
(201) 874-6833	(303) 759-2625	(415) 661-0705	(714) 537-7913
(201) 891-7441	**(303) 789-0936**	**(415) 683-4703**	**(714) 565-0961**
(201) 968-1074	(305) 261-3639	(415) 792-8406	(714) 571-5550
(202) 337-4694	**(305) 566-0805**	**(415) 851-3453**	**(714) 582-9557**
(202) 635-5730	(305) 689-3234	(415) 948-1474	(714) 730-1206
(203) 348-6353	**(305) 772-4444**	**(417) 862-7852**	**(714) 739-0711**
(203) 357-1920	(305) 821-7401	(419) 865-1584	(714) 751-1422
(203) 746-4644	**(305) 989-9647**	**(502) 245-8288**	**(714) 772-8868**
(205) 945-1489	(309) 688-0470	(502) 896-9624	(714) 898-1984
(206) 244-5438	**(309) 694-6531**	**(503) 646-5510**	**(714) 952-2110**
(206) 246-8983	(312) 255-6489	(512) 657-0779	(714) 962-7979
(206) 482-5134	**(312) 269-8083**	**(513) 671-2753**	**(714) 963-7222**
(206) 482-5590	(312) 337-6631	(513) 874-2283	(801) 375-7000
(206) 524-0203	**(312) 420-7995**	**(515) 279-8863**	**(801) 466-1737**
(206) 546-6239	(312) 622-8802	(516) 938-9043	(801) 753-6800
(206) 723-3282	**(312) 528-7141**	**(523) 223-3672**	**(802) 748-9089**
(206) 937-0444	(312) 545-8086	(602) 866-0258	(802) 879-4981
(209) 638-6392	**(312) 622-9609**	**(602) 956-5612**	**(803) 270-5372**
(212) 245-4363	(312) 767-0202	(602) 957-4428	(803) 270-5392
(212) 448-6576	**(312) 782-8180**	**(602) 957-9282**	**(803) 279-5392**
(212) 787-5520	(312) 782-9751	(604) 687-2640	(803) 771-0922
(212) 997-2186	**(312) 941-9009**	**(607) 754-5571**	**(803) 772-1592**
(213) 276-4276	(312) 964-7768	(607) 797-6416	(804) 340-5246
(213) 316-5706	**(313) 288-0335**	**(609) 983-5970**	**(805) 484-9904**
(213) 329-3715	(313) 357-1422	(612) 561-6311	(805) 527-9321
(213) 340-0135	**(313) 465-9531**	**(612) 929-8966**	**(805) 682-7876**
(213) 346-1849	(313) 477-4471	(614) 272-2759	(805) 964-4115
(213) 349-5728	**(313) 484-0732**	**(614) 649-7097**	**(806) 355-5610**
(213) 360-6332	(313) 569-2063	(615) 254-9193	(813) 223-7688
(213) 394-1505	**(313) 588-7054**	**(617) 354-4682**	**(816) 523-9121**
(213) 395-1592	(314) 838-7784	(617) 388-5125	(816) 531-1050
(213) 396-3905	**(316) 746-2078**	**(617) 431-1699**	**(816) 861-7040**
(213) 424-3506	(319) 353-6528	(617) 649-7097	(816) 931-3135
(213) 428-4718	**(319) 557-9618**	**(617) 692-3973**	**(817) 855-3916**
(213) 459-3177	(404) 394-4220	(617) 864-3819	(817) 855-3918
(213) 459-6400	**(404) 733-3461**	**(617) 897-0346**	**(817) 923-0009**
(213) 566-8035	(404) 790-8614	(617) 963-8310	(901) 276-8196
(213) 631-3186	**(404) 793-1045**	**(702) 826-7234**	**(901) 362-2222**
(213) 657-8803	(404) 939-1520	(702) 873-9491	(901) 761-4743
(213) 673-2206	**(404) 939-8429**	**(703) 281-2125**	**(902) 794-8198**
(213) 675-8803	(404) 953-0723	(703) 281-2222	(904) 243-1257
(213) 709-5423	**(405) 353-2554**	**(703) 379-0303**	**(904) 243-8565**
(213) 795-3788	(405) 528-8009	(703) 620-4990	(913) 362-6398
(213) 787-4004	**(408) 241-1956**	**(703) 734-1387**	**(913) 764-1520**
(213) 799-1632	(408) 296-5799	(703) 750-0930	(913) 782-5115
(213) 799-6514	**(408) 263-0248**	**(703) 893-9474**	**(915) 584-5393**
(213) 826-0325	(408) 263-9650	(703) 978-7561	(916) 393-4459
(213) 828-3400	**(414) 241-5406**	**(707) 448-9055**	**(918) 224-5347**
(213) 843-5390			

A SPACESHIP SIMULATOR

by David J. Beard

"I christen thee Miss Calculation"

Photos by David J. Breard

Imagine this scenario: you're sitting in the cockpit of a single-seat, deep space interceptor watching a vector display tick off the progress of a homing torpedo dropping down on you from directly above. Your seat vibrates in time to the rasping, low-pitched hum of your engines as you drive upwards to close. At about 20 kilometers, you punch a quick calculation into the nav computer: 12 seconds to slew to a reciprocal course; 13.2 kliks at a closing speed of 2200 meters/second. You enter the command "SLEW 135, 200" and hit the executive as the torpedo's range drops past 13 kliks.

With a clashing of gears, the ship begins gimballing. Stars slide across the viewscreen, followed in a few seconds by the spider-like shape of the torpedo rushing through the space you occupied six seconds before. The torpedo overshoots badly; as you complete the 200 grad turn, it's almost 10 kliks away, receding, again directly overhead. At 15 kliks, it slows and begins closing again; no overshoot this time. Again, a rapid set of calculations: final closing speed ought to be about 50 m/s; effective range on your laser is about 1100 m; call it 1700.

Your hand hovers over the weapons panel as the torpedo drops downward, slowing rapidly. At 1700 meters you tap a key, "CALL TARGET." More whirring gears as seconds pass. The torpedo swings into your field of vision at 129 mark 45, only 1100 meters out. Did you wait too long? Tap a key. "SINK ON;" with a barely audible mosquito whine, the heat sink is activated. 1000 meters, 129 mark 12. "LASER ON;" dashed tracers converge on empty space, swinging rapidly towards the torpedo. Finally, at 950 meters, the "ON TARGET" indicator lights.

The whirring of gears becomes intermittent as your ship automatically tracks the target. Range shrinks through 900 and 800; "DETONATED." Sirens whoop and warning lights blink into life, "TAKING FIRE;" "LASER HOT." The spidery shape on your screen is now a round fireball. The range continues to shrink and your sink temperature climbs. Suddenly there's a loud "clank" from forward and a muted explosion behind your head. The whooping siren picks up a beat, and more lights wink on. Too many are red! The cockpit is filling with smoke. A large boxed legend on the viewscreen reads "PREPARE FOR IMMEDIATE EJECTION." Another loud clank; with a last strangled whoop from the siren, everything goes dead. As the ventilating fans sigh to a stop, you're left sitting in darkness and silence in a smoke-filled cockpit. Now you know how all those Klingons felt about you and Captain Kirk.

"Miss Calculation," the simulator used to accomplish all this, is not a terribly complicated machine, although years went into the planning of it. The story starts back in 1972 with a home-brew analog computer outputting to an ancient high-persistance Dumont oscilloscope. That machine wasn't much of a spaceship simulator, but it did teach me a lot about working with differentials. In late 1975 the micro revolution broke and a year later, after many phone calls to Utah, I owned a genuine, working (mostly) Sphere System 3. The first applica-

David J. Beard, R.D. #1 Box 648, Newmanstown, PA 17073.

tion I programmed for it was a simulation called FIGHTER-TRAINER. Written in integer BASIC, this program used essentially the same scenario I described above; the trig functions were accurate to 2%, there was no interrupt driven I/O, and update time was about 20 seconds, but it ran. This early version had no rotary transforms; that is, a direction was always referenced to some arbitrary set of axes – the universe did not revolve around you when you turned your ship. FIGHTER-TRAINER was actually distributed by Programma, and I understand that a few copies were sold. In April of 1977 we Sphere owners finally got a floating point BASIC (still no trig functions, though). This improved the precision of FIGHTER-TRAINER amazingly, but did nothing for its speed.

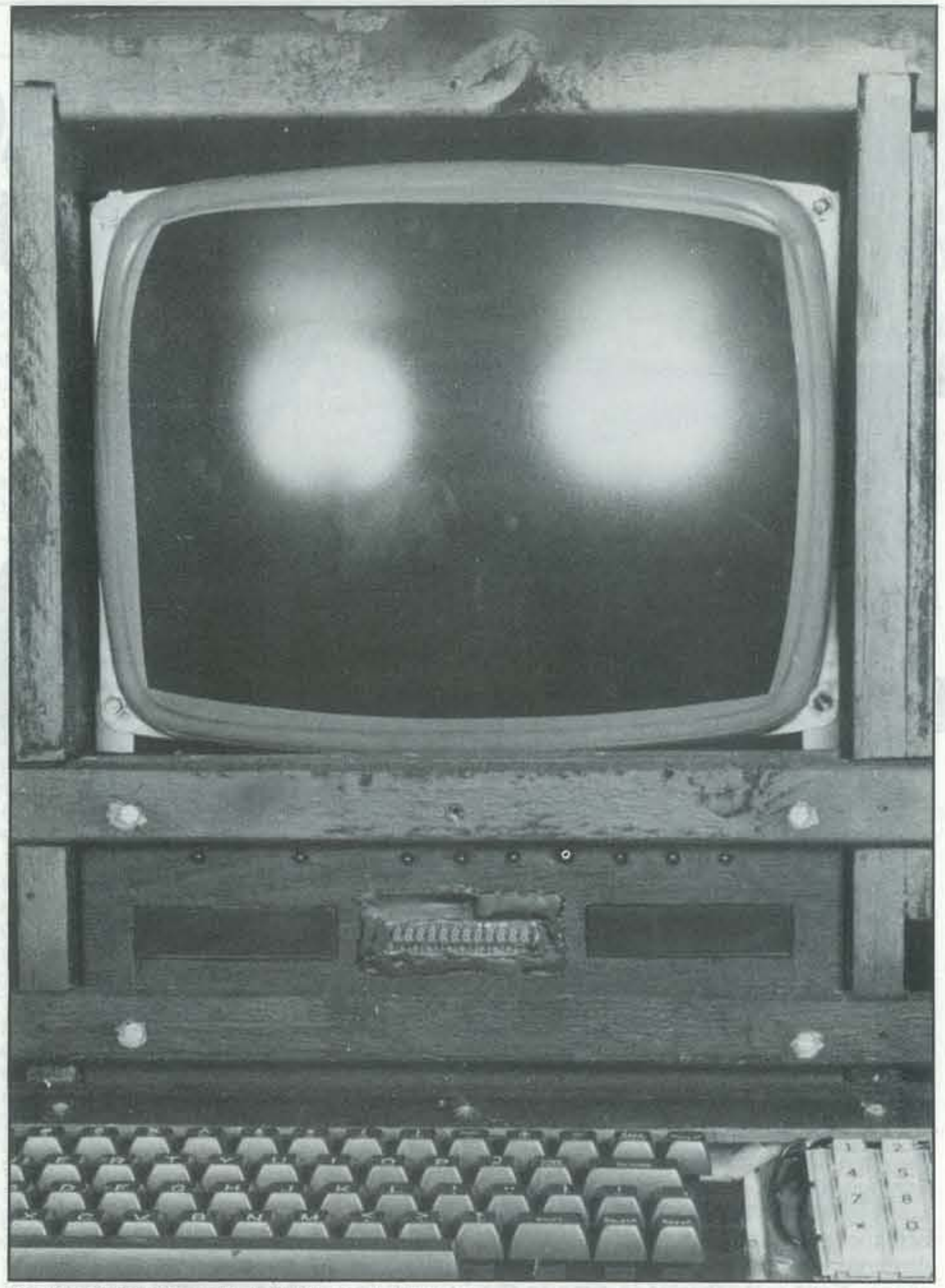
In June 1977 I did a program called SPUTNIK which simulated near-Earth orbits. It would maintain an orbit as low as 200 km for five or six orbits, but it was deathly slow. An orbit took 35 minutes, about a third real time. That's probably better than the Soviets could do in 1957, and it was kind of fun to follow the plots with a globe, but it was becoming obvious that BASIC was just too slow. I also tested rotary transform routines around this time. It turned out that if gravity wells and rotary transforms were built into a simulator program, I could expect update times of 90 to 120 seconds! I needed to increase speed by two orders of magnitude, and I wanted to get rid of that "wait for input" loop and run interrupt driven I/O.

In spring of 1978 I wrote QUAD, a QUick And Dirty floating point package. Conceptually, QUAD is a software model of a 5" sliderule. It uses 4-place, look-up tables for logs, log sines, and log tangents. Numbers are stored internally as 4-digit BCD 10's complement numbers with 2-digit decimal exponents. Multiplication and division are done by adding logarithms, exponentiation by a simple BCD multiply. QUAD is fast. The loop,

```
FOR A = 1 TO 10000
C = A * SIN(B)
NEXT A
```

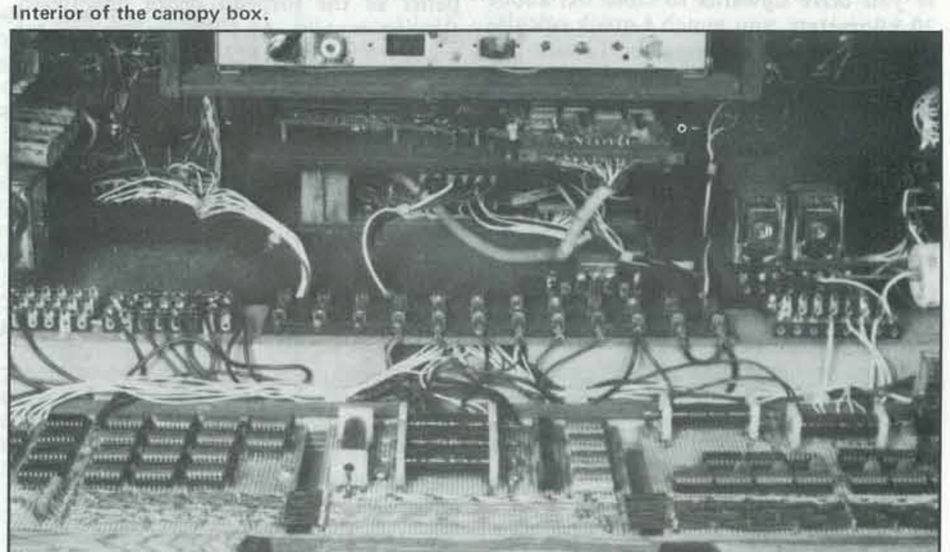
executes in 20 to 30 seconds, including the overhead for assignments and looping. It also has fairly fancy I/O routines. The floating point parser has fewer quirks than any I've worked with, and the output routine uses only exponents that are multiples of three; in other words, you get millimeters, meters, or kilometers, but not decimeters or hectameters (this is a fairly cheap feature to implement, and ought to be used more widely in engineering packages).

In June of 1978, QUAD was tested on a one-dimensional real-time lunar lander. This was the first real-time simulation I had used; the increased difficulty was startling (it isn't so much the difference in speed – it's the panicky feeling that things are getting away from you). In



Display panel with smoked plastic cover removed.

Interior of the canopy box.



any case, I now had a computer that "waited for no man."

At about this same time, two friends of mine were building a simulator – Lance Strickler and John Buffington of 2005 AD, Inc., in Philadelphia. This was the "SFS Walletsize" which was displayed at several trade shows. The Walletsize was roughly a 4-foot cube with one open side. The pilot's seat slid into the open side on a track, and the other 5 sides were studded with speakers, blinking lights, CRT's and controls. I was in their shop when they tested a real-time 2-D lunar lander simulation. None of us were very good at it; it looked more like we were strafing the moonbase than trying to land on it (the theme music from Star Wars booms heroically in the background, and a kibitzer calls, "Run for it, Charley! Here he comes again!"). After that afternoon I knew I had to have my own spaceship.

The Walletsize was designed expressly for trade shows. Simulations were kept short and simple so they could be demonstrated to a large number of people, and the special effects were aimed as much at spectators as at the operator. Also, the Walletsize was expensive. No fewer than three computer systems were cannibalized to put it together. At one point Buff remarked, "When I was a kid I took pinball machines apart to make computers. Now I'm taking computers apart to make a pinball machine." I was primarily interested in teaching myself to fly a spaceship, and I had a limited budget to work with. During July and August of 1978, my plans for "Miss Calculation" fell into shape.

These were the criteria I considered: all existing software ought to run on the Sphere with the simulator in place, and it should be possible to use the simulator without physically disturbing my existing system. There was already coax and a 15-wire parallel bus running around my shop. The simulator would have to use these lines.

The simulator ought to be completely enclosed and as nearly light- and sound-proof as possible to minimize distractions for the operator. Special effects should be realistic enough to provide subconscious cues (the operator should not have to scan a screen full of fine print to learn that his engines have failed!). They should also be simple to operate at the assembly level. The interface between the simulating program and the simulator would ideally be a set of single-bit flags.

All of the operator input should go through a keyboard. I believe that joysticks are as out of place on a spaceship as a buggy whip on an automobile. Ideally, the only input device would be a microphone, but voice I/O isn't that good yet. Also, all functions should be under software control rather than hardwired to switches with sense lines. This makes for more expensive interfacing and a less

Figure 1

CONTROL FUNCTIONS AVAILABLE FOR MISS CALCULATION

TOGGLE FUNCTIONS

Interior Lighting
Air Conditioning (fan and panel indicator)
Subspace Radio (intercom and panel indicator)
Battle Computer (timer, calculator, keyport indicators)

SET-RESET FUNCTIONS

Laser (piezo buzzer and panel light)
On Target indicator
Laser Hot indicator
Heat Sink (piezo buzzer and indicator)
Taking Fire indicator
Sink Hot indicator
Main Drives (engines, indicator, "fuel flow" indicator)
Gimballing Gyros (gyros, indicator, "fuel flow" indicator)
Yellow Alert (siren, LED and 110V lamps)

MOMENTARY FUNCTIONS

Reset (turn off all functions, zero timers)
Tick (advance clock and resettable timer)
Crash (red alert lamps, siren to high speed, "power" lamp off, "power failure" lamp on, "auxiliary power" lamp on, smoke generator on, rising edge sets "crash" relay, falling edge resets "crash," kills all power)

HARDWIRED CONTROLS

Push-To-Talk Key ("break" key on keyboard)
Reset Timer ("repeat" key on keyboard)
Start-Run-Stop Switch (outside canopy on connector panel)

Figure 2

SUMMARY OF INSTRUCTIONS FOR FIGHTER TRAINER

LIGHT – Turn interior lights on or off

AIR – Air conditioning on or off

RADIO – Radio on or off

COMP – Battle Computer on or off

JETS (OFF) – Main Drives on (off)

LASER (OFF) – Laser on (off)

SINK (OFF) – Heat sink on (off)

CALL – Call target. Ship will gimbal automatically to hold target at declination zero.

RELEASE – Release target. Countermands "CALL" instruction.

SLEW (azimuth),(declination) – Rotate tail of ship towards (azimuth) for (declination) grads.

(**CALL, RELEASE,** and the **LASER** and **SINK** commands are available as single special function keys on the Weapons Control Panel.)

colorful control panel, but it allows the computer to simulate failures or obey complex user-defined commands.

The simulator ought to have dual controls so that it could be used for instruction. I wanted very much to make a two seater, but it became obvious that the second seat more nearly quadrupled the cost of the finished simulator than doubled it. I settled for a second control station at the Sphere console.

The simulator had to fit the floor space available in my shop and fit through the outer door. And finally, it had to fit within my budget.

Meeting that last condition was touch and go. When it came down to a choice between the spaceship and a second family car, the spaceship won. In late August I took some drawings and a graph paper model to Glenn Smith, a carpenter friend of mine. A week later, we unloaded the Miss Calculation from his pickup and brought it into the shop. Glenn said he had a lot of fun building it. People would ask, "What are you working on this week?" and he'd say, "Why, I'm building a spaceship for Dave

Beard." "Oh Yeah? When is he leaving?"

It's essentially a cradle-shaped box surrounded by a flat deck. The back of the cradle rises around the head and shoulders of the operator, and a sliding canopy comes back to complete the enclosure. Cables connect the sliding canopy box to the wall and to the lower part of the simulator. The canopy contains all operator controls and displays, and the bulk of the electronics.

In the interior are the function decoder boards. To the left are power supplies and the display module. The video monitor is supported by a cradle hung from the firewall. The "battle computer" is just visible below it. To the right is crash-reset and alert circuitry.

The left side of the firewall is all alert displays; the right is all idiot lights with legends. Below the video monitor is the "battle computer." It consists of two counters used for clocks (one is a reset-table timer), a Radio Shack scientific calculator, and nine LED's that display the bit code on the keyboard port. The calculator keyboard is to the extreme lower right, and is hardwired directly to the calculator. The white touchtone keypad is wire-or'ed with the ASCII keyboard, but has a different strobe line. Two optical spotlights illuminate the keyboard when the interior lighting is off.

The rear of the simulator has two access panels. The "engines" are mounted on the back of the pilot's seat. A heavy speaker is fed a half-wave, rectified AC to simulate the main drives. The speaker is spaced away from the plywood by rubber washers, and can be tuned to produce satisfactory sub-sonics. The "gimballing gyros" consist of a plastic ring gear driven by three small plastic trolleys. Above the deck are two small light bulbs used for indirect interior lighting, a fan used for ventilation, and a transformer for the smoke gnerator. To the right of the pilot's head is the "subspace radio," made from two Star Trek walkie-talkies.

Hold your horses, can't ya? I think I know where the Dungeon Doorway is!

The Miss Calculation was completed by May of 1979, and I decided to shake down the system by writing a simple 3-D lunar lander program called "FLATLANDER," using cylindrical coordinates and still without rotary transforms. This "simple" program turned into a 16K monster of a freight-hauling simulation, with elaborate communications between Miss Calculation and the "Port Authority," a balance sheet updated in real time to itemize costs and give a figure for dollars/ton-km, and an interactive "ship's library." After maintenance, fueling, and loading, the pilot is responsible for power-up procedures, obtaining proper clearance, calculating a trajectory, takeoff and landing. Range and gross vehicle weight vary over a wide range.

Typical time for a FLATLANDER mission is 10 to 25 minutes, of which at least half is spent on approach and landing. For what it's worth, it is humanly possible to land a heavy ship on its jets without computer guidance, but it requires painstaking concentration, wastes a lot of fuel, and leaves very little margin for error or inattention. As the time required to slew the ship away from the vertical and back increases, the exercise becomes nearly impossible. If it were absolutely necessary to land a large ship manually, it would probably be best to use a three-man crew organized as pilot, helmsman, and engineer.

Finally, in summer of 1980, I wrote FIGHTER TRAINER 3.0, the simulation I had been envisioning all along. FIGHTER TRAINER uses spherical coordinates and permits true rotary transforms. It is a duel between a single ship and a single homing torpedo, a game of energy transfer and heat dissipation. The physics are as nearly correct as the low-precision mathematics will permit, and the capabilities of the ship and the torpedo have been carefully matched to allow a reasonable chance of failure or success in a wide range of maneuvers.

Early BASIC versions of this simulation gave coordinates in degrees, with declination measured from the azimuth. All of the recent versions use grads instead of degrees. The pilot must be able to calculate reciprocal and normal vectors rapidly, mentally. I found that even after considerable practice, it was easy to make mistakes when mentally adding 90 or 180 degrees to an angle. Adding 100 or 200 grads is much more foolproof. Also, I now measure declination from the nadir

rather than from the azimuth. That is, 0 mark 0 is straight down, 0 mark 100 is straight forward (north), and 0 mark 200 is straight up. The azimuth is measured clockwise from the north, as in geography, rather than counterclockwise from the east as in physics. In space combat, if you are trying to close with an object it will be straight up. When you get halfway there, you have to flip over and accelerate away from the object to brake your velocity. Now it is straight down. It turns out that most of the things you are interested in will be straight down, which is why declination is measured from the nadir.

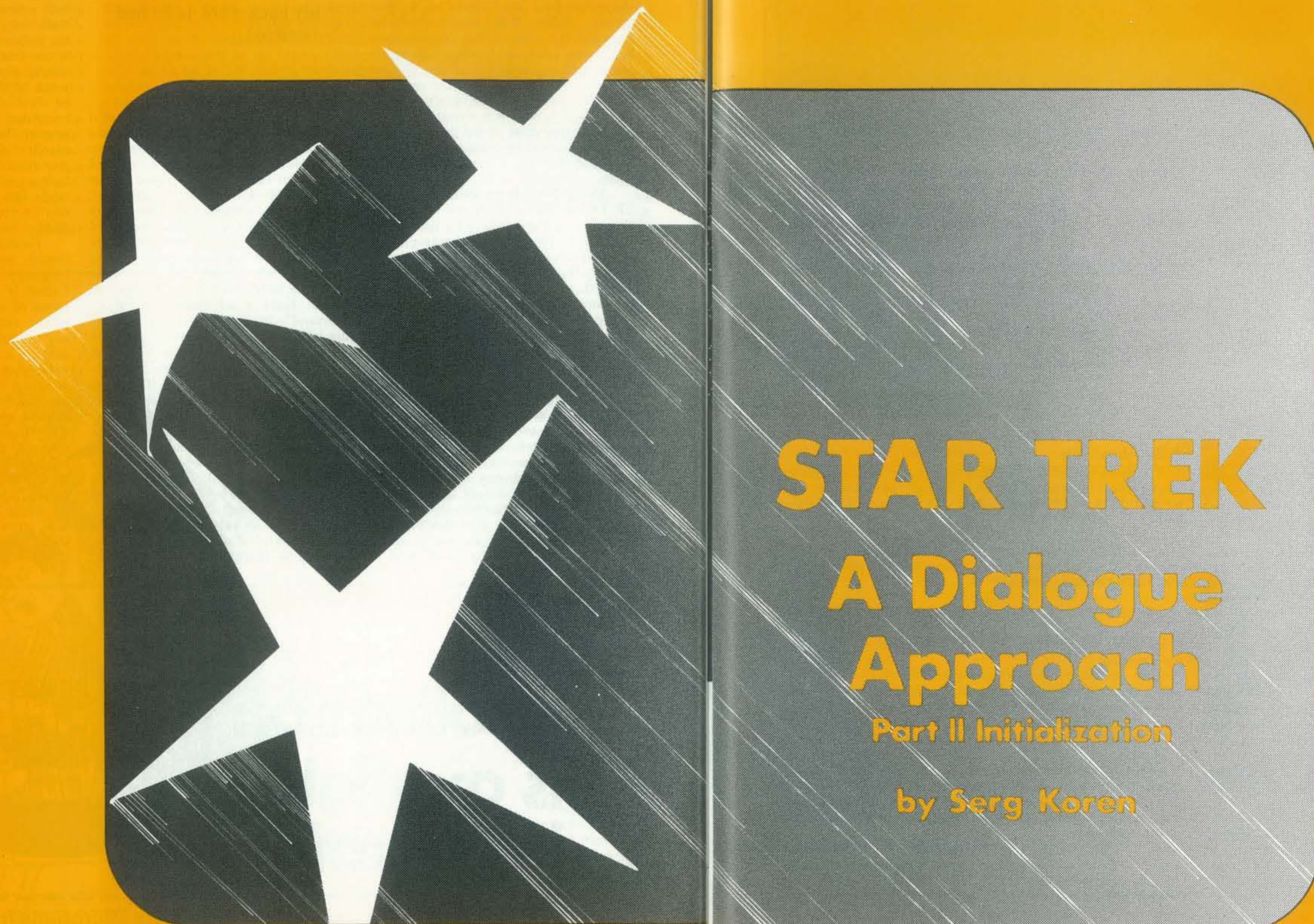
The ship being simulated in FIGHTER TRAINER is a single seat interceptor about the size and shape of a lunar landing module. The engines and sensor arrays all point straight down. (Which way is the target? Down!) It's equipped with an active heat sink (roughly equivalent to the "shields" in Star Trek) that increases heat dissipation by an order of magnitude when it's operating. It can slew (pitch) in any direction at a rate of 8 grads per half second. There are only two throttle settings, on and off. Acceleration is 10 m/sec/sec continuous. The homing torpedo is smaller, accelerates at 50 m/sec/sec, and slews at over 200 grads per half second. When detonated, it burns for 16 half seconds without accelerating.

The starfield display is a background and moves at the proper speed in the proper direction as the ship slews, but it is generated randomly at the leading edge of the field – there's no attempt to rotate the night sky in real time! The pilot is given bearing and heading vectors and sink and laser temperatures (the laser can overheat and kick out on thermal overload). All on-off indicators are handled by the idiot lights, except for the missile status word ("inbound," "targeted," "detonated," "destroyed"). The display is updated twice a second. Keyboard input is interrupt driven, and appears immediately in a command buffer at the bottom of the screen; however, commands are only processed once each half second. There is no "break" or "quit" command – to quit, you climb out and turn off the simulator. There is no scorekeeping. Torpedoes continue to appear one at a time until one gets through. When the ship is "destroyed" it turns itself off and must be reset from outside the simulator. A torpedo may be destroyed by laser fire before it detonates. After detonating, the resulting fireball may or may not deliver enough energy to overload the heat sink and destroy the ship – this depends mostly on the relative speed at the moment it detonated.

Any UFO that lands in my back yard is in bad trouble!

The Miss Calculation is not a commercial product. The hardware is unique, the software is highly machine-dependent, and documentation is sketchy. I am not prepared to distribute plans, schematics, or listings. I do hope that this article will inspire someone else to do a better job with newer technology. At this time the Walletsize and the Miss Calculation are the only privately owned spaceship simulators that I'm aware of, and both are several years old. It's worth noting that much of the expense of a simulator can be eliminated by mounting an appliance computer directly in the simulator. I suspect that an Apple II and Sublogic's excellent flight simulator package would give someone a flying start. If you have built or are building a simulator, I'd certainly like to hear about it – but please don't write to me. Write it up for publication!

Finally, was it worth all the time and trouble and expense to practice evading imaginary torpedoes in deep space? Yes, I think it was. People normally think in terms of cylindrical space (why does a mirror turn things left for right, but not upside down?). We are very dependent on a local up-down reference, and very upset when we don't have one. The conquest of space is going to require a psychological revolution as well as an industrial revolution. Real spaceships are still astronomically expensive, but with relatively modest equipment all of us can help to hammer out the new concepts we'll need to go with the hardware. Personally, I want to be sure that if I ever get my hands on a spaceship, anytime, anyhow, anywhere, I'll be able to fly it. Any UFO that lands in my back yard is in bad trouble! ■



STAR TREK

A Dialogue Approach

Part II Initialization

by Serg Koren

"These are the voyages of the Starship Enterprise..."

Editor's Note: This is the second of a three-part series on how to build a Star Trek program "universe" that will recognize "conversational" as opposed to "numeric" commands. Part I appeared in the July-August 1980 issue and covered the background.

"These are the voyages of the Starship 'Enterprise'. . ."

In the previous installment we dealt with the background of the computer program and the characters to be portrayed within it. The pseudo-language to be used was detailed and some hints as to how to implement the program were described. In this installment we will define, describe and initialize some of the computer program variables.

Initial Initialization and the Benefits Thereof

How often have you found yourself with a programming problem you had to solve? If you're like most humans with a computer, you have on occasion sat down in front of your keyboard and begun to hammer away at the keys improvisationally to write the program. This is fine if the program is short and simple, but when dealing with complex problems and simulations it is nearly impossible to write an efficient program in this manner.

A wise man once said, "Begin at the beginning." This concept should apply to all programming as well – the "beginning" in this case being initialization. If you sit down and just list all of the variables you need, assign them names, and determine their exact function and initial values, a lot of later hassle can be eliminated. This simple trick of initialization will automatically result in more readable and logical code because it eliminates redundant variables or otherwise unnecessary variables. This, in turn, produces more efficient code. Also, as you code, you have a ready list you can refer to when you need a specific variable. Although BASIC and most other languages do not require you to declare every variable in the program, there are many computer languages that do (TAL being one). The fact that the language you intend to implement doesn't require complete declaration shouldn't dissuade you from declaring every variable, if only on paper. The time you spend now can save frustration and confusion later.

Another benefit of coding initialization first is that if you own a small system you are often constrained by the limited number of variable names you are allowed (some versions of BASIC only allow one or two character names; as A, A1, B3$, etc.). By listing the function of each variable, it is possible to determine which variables can serve double or even multiple duty. If, say, you are using the variable "A" as a counter in one segment of the program, it is perfectly all right to use it as a non-counter somewhere else, if the two functions are isolated and the value of the two functions need not be saved. This simple trick allows you to increase the complexity of your program without the need for expanding the variable-name capabilities of the system.

Of course, with a really complex program it is very difficult to list *every* variable that is needed and some will tend to crop up in the actual coding phase. But these should be few in number once the main list has been made up.

Types of Variables and Varying Types

Integers, strings, floating point, fixed point, literals, direct, indirect, defines and even "plits" (pointer to a literal) exist in computerdom and I'm sure there are more types of variables. You will need to tailor the initialization list given in this installment to suite the types of variables used by your particular system. The list provided will use strings (alphanumeric quantities), integers (numeric quantities with an implied decimal after the rightmost digit), real values (numeric quantities with an explicit decimal) and pointers that are either string or integer in type. Pointers will be defined and described later. When converting to your system, remember that more than likely integers will take up the least memory and strings the most. Therefore, define the integers first and the strings last, because strings can be truncated whereas integers cannot.

A Bit About Bits

Some of the integer variables are actually bit arrays (or bit maps), where each bit of the word is essentially a unique variable that can have two states, either 0 or 1. Theses are declared by

X.(Y:Z)

where X is the integer name, Y is the lower bound of the bit map and Z is the upper. Hence, on an 8-bit processor, a full word would be designated as

X.(0:7)

whereas in a 16-bit machine it would be

X.(0:15)

A particular bit is designated by enclosing it in .(). Therefore, the statement

X.(4)=1

would result in the variable X being set to binary 00 001 000, or %10 (octal 10), or decimal eight in an 8-bit processor, and binary 0 000 100 000 000 000, or %4000, or decimal 2048 in a 16-bit machine. Note that in both cases bit 0 is the leftmost bit.

Because most home computers nowadays are 8-bit machines, the bit arrays in the initialization list will all be based on an 8-bit word. If your machine cannot handle individual bits in the language you code in, you will have to convert the bit arrays into integer arrays.

x.(0:7) will become X(0,7)

where (,) indicates an integer array. Of course, this increases the memory used. Instead of using 8 bits (one word), X(0,7) uses 64 bits (or eight words). As you can see, bit handling is much more efficient for flags and certain arrays than word handling.

The List

The initialization list that follows consists of the variable name, the array size (if it's an array), and its function within the program. To ease conversion into BASIC, the convention of appending a "$" to a variable name to denote a string variable will be maintained.

A. INTEGERS

- **ENEMY'UNIT'INDICATOR** – Indicates the "nth" enemy unit.
- **STARBASES'IN'QUADRANT** – The number of starbases in the quadrant where the ship is located. Quadrants defined later.
- **COMMAND'COUNT** – The length of the command input by the user.
- **DIFFICULTY'FACTOR** – How difficult a game are we going to play?
- **DAY** – Current day of the month.
- **MONTH** – Current month.
- **YEAR** – Current year.
- **ENERGY'OFFSET'FOR'ENEMY** – Different enemies have different minimum energies. This is that minimum. This is initialized programmatically.
- **MAXIMUM'ENEMY'DENSITY** – The maximum number of enemy ships allowed in quadrant. Different enemy types have different offsets.
- **ENEMY'COUNT** – The length of the enemy's name. For example, "KLINGON" is seven.
- **NUMBER'OF'ENEMY** – The total number of enemy units in the current game. This is programmatically set.
- **HOURS** – Current hour.
- **MINUTES** – Minutes after the hour.
- **SECONDS** – Seconds after the minute. The time and data may be obtained from an on-board clock, or input by the user.
- **MAXIMUM'ENERGY** – The maximum number of enemy units in the current game.
- **NAME'COUNT** – Number of letters in the Captain's (user's) name.
- **PHOTON'TORPODOES** – The number of photon torpedoes the "Enterprise" currently has. Initialize this to 30.
- **PROBES** – These are long-range sensor probes. Initialize this to three.
- **HOUR'LIMIT** – Omit if no on-board clock. This is the hour when the game ends.
- **MINUTE'LIMIT** – Omit if no on-board clock.
- **ALERT'STATUS'FLAG** – This is a three-state flag indicating ship's status.
- **DAYS'LEFT** – This is the subjective number of days left in the game.
- **PREVIOUS'DAYS'LEFT** – Holds DAYS'LEFT's last value.
- **STARS** – The number of stars in the game.
- **STARBASES'IN'GAME** – The number of starbases in the game.

B. REAL

- **ENERGY'PER'ENEMY'UNIT(1,200)** – A 200 element array that contains the total energy (power) that a certain ship has. Element one contains the energy of enemy unity one, etc. Initialize each element to 1000.
- **DILITHIUM'CRYSTALS** – The energy available to the "Enterprise" for manuevering, computers, sensors, etc. Initialize this to 1000.
- **REAR'PHASERS** – The energy available to the "Enterprise's" rear weaponry. Set this to 1000.
- **FORWARD'PHASERS** – Set this to 1000 also.
- **TOTAL'PHASERS** – This should always equal REAR'-PHASERS + FORWARD'PHASERS.
- **HEADING'XY** – The heading of the "Enterprise" in a horizontal plane. Most S.T. programs use 0 through 7 as headings. This will be used as explained in a future installment.
- **HEADING'YZ** – The heading of the "Enterprise" in a vertical plane.
- **SHIELDS** – The energy available to the ship for defensive purposes. This should be initialized to 500.
- **STARDATE** – The time system used in the Star Trek universe.
- **WARP** – Relativistic velocity of the "Enterprise."
- **SUBLIGHT** – Velocity of the ship below the speed of light. Both WARP and SUBLIGHT should be initialized to 0.

C. BIT MAPS

- **STARS'LOCATION** – Map where a set bit (1) indicates the location of a star.
- **STARBASES'LOCATION** – Location of starbases.
- **ENEMY'LOCATION** – Location of enemy vessels.

The size of these three arrays will depend on how big a universe you want, and the space you have available. A good starting point would be (0,9,9) (these are three-dimensional arrays). This would result in 8000 discrete points or locations in the universe (as opposed to 64 in a typical S.T. program). If your system can't handle three (or even two) dimensions, never fear! We'll discuss conversion techniques for going from one to "n" dimensions in the installment on navigation.

- **FLAGS(0,5)** – We'll define these as we need them.

D. STRINGS

- **NAME$** – User's name (Captain's name).
- **ENEMY$** – Name of enemy. We use "KLINGON," "GORN," "ROMULAN," and "V'GER." (This should be the first published reference to a game based on the movie, although I already have a program like this.)
- **COMMAND$** – User's command.
- **MISC$** – Used for anything else that isn't hardcoded.

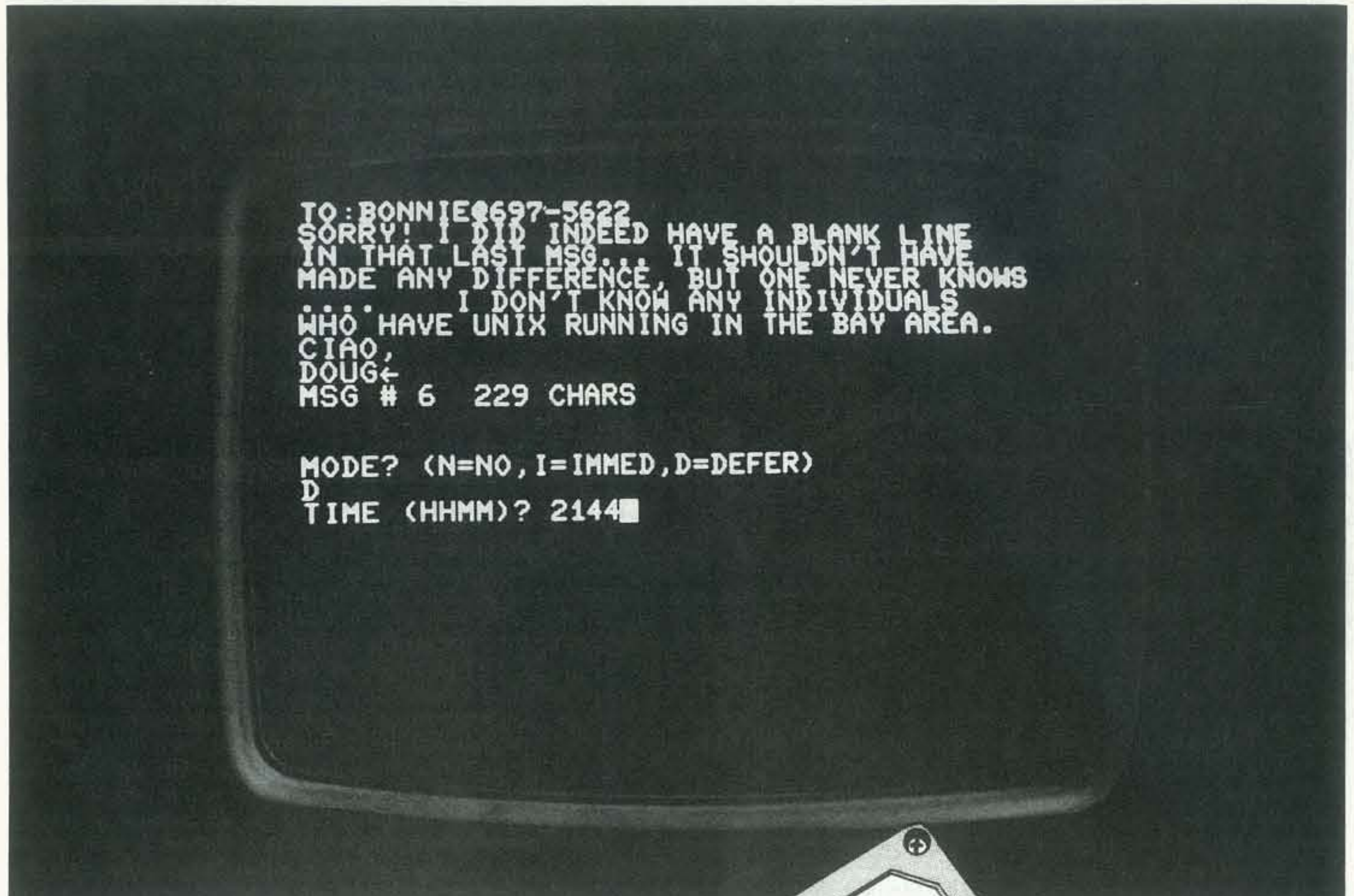
With this list it is possible to code a very complex program. Of course, as we develop the code, we'll probably think of more variables that we may need or want to use.

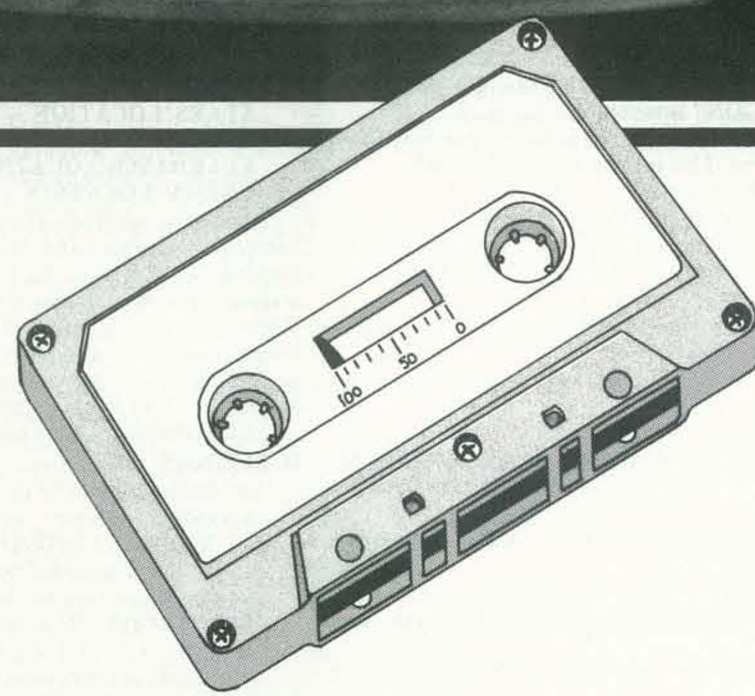
NOTE: The apostrophes in the variable name are equivalent to spaces and are only used to indicate that the name is actually a single entity.

Next Time

In the next installment most of the preliminaries will be coded in pseudo-language; initialization, as well as the theory and pseudo-code for the MISSION module. We'll also discuss and give an example of a relatively new and very powerful technique of structural flowcharting.

"live long and prosper" ■





Personal **C**omputing **N**E**T**work

PAN

Personal Computing NETwork (PCNET), a project of People's Computer Company, has available computer mail support software for the Commodore PET®. Other versions (including Apple) will be available shortly.

The new PCNET computer mail system is called PAN – a program on cassette tape for use with an 8K or larger PET. All that is required is a telephone line, an auto dial – auto answer modem, and a personal computer (available at present only for the PET).

The PAN software and a user's manual sell for $18; a user's manual is available separately for $3.

If you would like more information on the PCNET project, send a large, stamped (for 2 ounces), self-addressed envelope to PCNET, People's Computer Company, Post Office Box E, Menlo Park, California 94025.

Product News

by Dave Cortesi

What a Release!

Press releases! Drifts of them on the editor's desk, more in an untidy lump in a drawer. Any company with an announcement to make sends a press release to the editor of every magazine its PR department can think of. They come in modest single sheets and glossy brochures, all written in bright, upbeat journalese that the press agents think we'll use without re-writing. It's fun to read them, and they help us keep informed, but they present a serious editorial problem.

There are nuggets of good information buried in the press release pile; many of the announcements tell of things we think you'd like to know. But the releases are written in a self-serving way. That's only natural; if you could write a news item about yourself, you'd make it a favorable report, wouldn't you? There's rarely anything false in a press release, but they do lack perspective, balance, context. Most magazines print them straight or lightly cut, and we've done that too. But we're going to try a different approach this time.

In this column we're going to sift out what we think are the most interesting releases, rewrite them, juxtapose them in useful ways, and make occasional comments. With some hard work, the missing perspective and balance will emerge. What's more, we're inviting you to supply even more of those things! Where we end an item with **(experiences wanted)**, we mean just that. That phrase marks new products about which we'd like to publish accounts of our readers' experiences. If you try one of these products, and if you'd like to have your name in print, please write to us. Name the product; tell us when you'll have it, what kind of use you'll put it to; give an indication of how thorough a review you're capable of writing. We'll get right back to you!

Learn, or Teach, BASIC Programming

This month's pile yields three new ways to learn programming in BASIC. Radio Shack announced "Introduction to BASIC Programming, Part I," a package "designed to provide students with a first experience in programming" which Radio Shack claims requires "little programming or computer knowledge on the part of the instructor," a welcome note indeed. The package includes a teacher's manual, overhead transparencies, and student workbooks. One or more TRS-80s (of course) are needed for the hands-on work; the package costs $160 **(teacher's experiences wanted)**.

Two publishers have announced books for the independent student of programming. Sybex (2344 Sixth St., Berkeley, CA 94710) has two: *Inside BASIC Games* by R. Mateosian uses successively more complex game programs as examples of program design and coding; *Fifty BASIC Exercises* by J. P. Lamoitier presents the language through exercises accompanied by flowcharts and comments. The books are priced around $14; Sybex didn't indicate what flavor of machine they are aimed at. Metra Instruments Inc. (2056 Bering Drive, San Jose, CA 95131, 408-297-8530), on the other hand, have four different version of their book *Training Your Computer*, one each for the Apple II, Compucolor, Pet and TRS-80. Each of the book's 64 pages is said to introduce a single BASIC concept or technique; the price is around $5 including handling and quantity discounts are available.

Those who are between 10 and 17, and find reading too dry, might consider mixing their programming with swimming and volleyball at a summer Computer Camp operated by the Grand Union Lodge (POB 22, Moodus, Connecticut) and managed by Dr. M. Zabinsky (203-795-9069). Two one-week sessions will be offered in July of 1981 **(reports wanted on this and similar camps)**.

Other Educational Software

We've a bumper crop of educational software this time. Metra Instruments (address above) is distributing a set of trigonometry-teaching programs for the Compucolor/Intecolor machines that provide "experiences with radian measure, the . . . sine function, . . . drill with identities, and polar graphs." The programs are the work of M. A. Fitting of San Jose State University; they emphasize the Compucolor's graphics. Programs, source code (good!) and 70pp manual cost $30; educators' discounts available **(teachers' experiences wanted)**.

Planning Cash Flow is the title of both a program and a self-study course. The package runs on an Apple II; it claims to teach how to get a picture of a company's cash flow, how to analyze it, and how to minimize the cost of short-term credit. A most timely subject with Educational Programming Systems (1328 Baur Blvd., St. Louis, MO 63132, 314-991-0300) will teach you for $100. The potential cost/benefit ratio, given present interest rates, is almost frightening **(experiences wanted)**.

The Blacksburg Group are publishers of a Continuing Education Series, one title in which is *Circuit Design Programs for the TRS-80* by H. M. Berlin. The book presents 40 programs that solve problems in circuit design and basic statistics. The same programs in machine-readable form are now available on cassette tape for the TRS-80, on tape or diskette for the Apple II and OSI machines. Book programs, and handling charges add up to about $40; contact Group Technology Ltd, POB 87s, Check, VA 24072, 703-651-3153.

Borg-Warner Educational Systems offers a Supplementary Reading Program based on the Apple II, which claims to emphasize "skills of inference, paragraph meaning, and general comprehension," and includes a progress reporting system. We've little information on this package, except that it occupies a total of eight diskettes and costs a rather startling $750 **(teachers' experiences wanted:** is that price justified?).

Finally, Serendipity Systems Inc. (225 Elmira Rd., Ithaca, NY 14850, 607-277-4889) has prepared an Instructor's Gradebook system for the Apple II. This $169 package claims to handle grades in a number of categories, each separately weighted, with scores enterable as numbers or as letter grades with user-defined values (a nice touch). The package is said to produce several useful-sounding reports and the standard statistical analyses **(experiences wanted** on this and similar packages).

General Software for CP/M Systems . . .

The Information Master is an intriguing program from Elliam Associates (24000 Bessemer Street, Woodland Hills, CA 91367). It is a text-retrieval system, a program that keeps a dictionary of up to 1500 words or phrases (user-defined). The program will pass over any text file, locating keyword occurrences and noting them in an index on disk. Thereafter text can retrieve from an indexed file by word or boolean combinations of words. If it works well, the program is a steal at $39.95; a pre-indexed file of "hundreds" of magazine references to CP/M costs $10 **(experiences wanted)**.

. . . for Heath Systems . . .

Heath, which has been working very hard to make more, and more useful, software available for their machines, has released a slug of programs under their "Softstuff" label. Some are CP/M packages already available from other sources, such as CBASIC and the BDS C compiler. Others, such as a business inventory program, a fast sort and a clever ham radio processor, are available only for Heath's HDOS operating system. A program to link your Heath computer to the Micro-Net timesharing service and a General Ledger package are available for both operating systems. Write to Heath Co. Dept. 350-670, Benton Harbor, MI 49022, for a free catalogue.

. . . for the TRS-80 . . .

A general accounting package for the TRS-80 Model II (not Level II, but the business system), including general ledger and accounts both receivable and payable, is available for just $55. Is there a catch? Yes and no. The package is said to be "modeled after the famous Osborne software," which as many people know is available as a book at half the price. On the other hand, it's probably worth $55 just to have had someone else key the program in for you; if they've been tested as well, the price is a bargain.

. . . for the Atari . . .

An outfit named Macrotronics (1125 N. Golden State Blvd., Suite G, Turlock, CA 95380, 209-667-2888) has jumped into the nascent Atari software market with both feet. Their Screen Printer package ($139) contains a connector that links a Trendcom or IDS printer to the Atari's number 3 and 4 jacks, and a machine language program to drive the printer. The program will copy the exact screen image including graphics to paper, and adds BASIC commands for printing data and programs **(experience wanted)**. The same company offers a Morse Code Tutorial designed to ready you for the FCC ham license tests.

. . . for the North Star . . .

Allen Ashley (395 Sierra Madre Villa, Pasadena, CA 91107, 213-793-5748) wants to make programs written in North Star BASIC run faster. To that end, he supplies a BASIC compiler. The compiler reads a source program in BASIC and writes a file of assembler language statements; these, assembled, yield a machine language program equivalent to the original, but larger and faster. The price is $400, which suggests that Allen expects to sell to professional programmers.

. . . and for the Apple II . . .

CompuSoCo (26251 Via Roble, POB 2325, Mission Viejo, CA 92690) is pleased to announce software packages for professionals who own an Apple II disk system. Each package includes a variety of billing, accounting, scheduling and time management functions. One package is customized for dentists, one for attorneys, and a third for "consultants and contract administrators." Each package costs $750 **(experiences wanted**, especially comparisons to similar packages).

If you aren't a professional and want to apply your Apple II disk system to more modest accounting work, you may be interested in "the easiest handling of checkbooks to date." That's the claim of The Computer Emporium (3711 Douglas Ave., Des Moines, IA 50310, 515-279-8861) for their CHECK-MATE package. It costs a modest $60, and lets you enter, edit, sort, and display a year's checks on a single diskette. CHECK-MATE (love the name) claims to help you reconcile your bank statement, examining your entries and displaying suspected errors when you can't balance **(experiences wanted**; we've always doubted the utility of checkbook programs).

Muse (330 N. Charles St., Baltimore, MD 21201, 301-659-7212) writes to tell us about a clutch of new Apple software: yet another word processor, a form letter program, a data plotter, each of which is said to be the ultimate of its kind. The announcement that grabs us tells of what sounds like a genuinely new idea in computer games. RobotWar operates two to four murderous battle robots on a game field. Each robot is independently programmed by one of the players, using a "battle code" reminiscent of the Big Trak toy of last year. When each player's robot is primed, they're all turned loose to fight it out, may the best programmer survive. This might be a loser or it might turn out to be deeper than chess **(experiences wanted)**.

A Few Hardware Items

Most press releases that talk about hardware are so esoteric that we hold them for *Dr. Dobb's Journal*; we assume that the average reader of *RC* isn't terribly interested in S-100 memory boards and the like (if we're wrong, let us know). This month's stack, however, yielded a few items of general interest. The first is a price-change announcement from APF Electronics. Effective January 1, 1981, the price of the APF Imagination Machine has been reduced from $599 to $399. This has been made possible by "technological advances [and] efficient manufacturing." All very well, but we can't resist noting the timing of the announcement; hope you didn't buy one at the old price for Christmas.

Mosaic Electronics (Box 748, Oregon City, OR 97045) expects to have a 32K memory board for the Atari ready for sale in February. They don't tell us the price, but they do point out that the board will fit not only the Atari 800 (for which Atari also sells a larger memory) but the Atari 400 as well (for which Atari does *not* provide a large memory). They claim that with their board, an Atari 400 will be able to use disk drives.

Connecticut MicroComputer Inc. (34 Del Mar Drive, Brookfield, CT 06804) sells a line of nicely-packaged devices that expand the powers of several popular home computers. With their products, an Apple, Pet, Kim or TRS-80 can be made to sense light or temperature, read analog voltages, or drive the BSR X-10 home controller. Ask for a catalog; add a sense to your computer **(experiences wanted)**.

Book Reviews

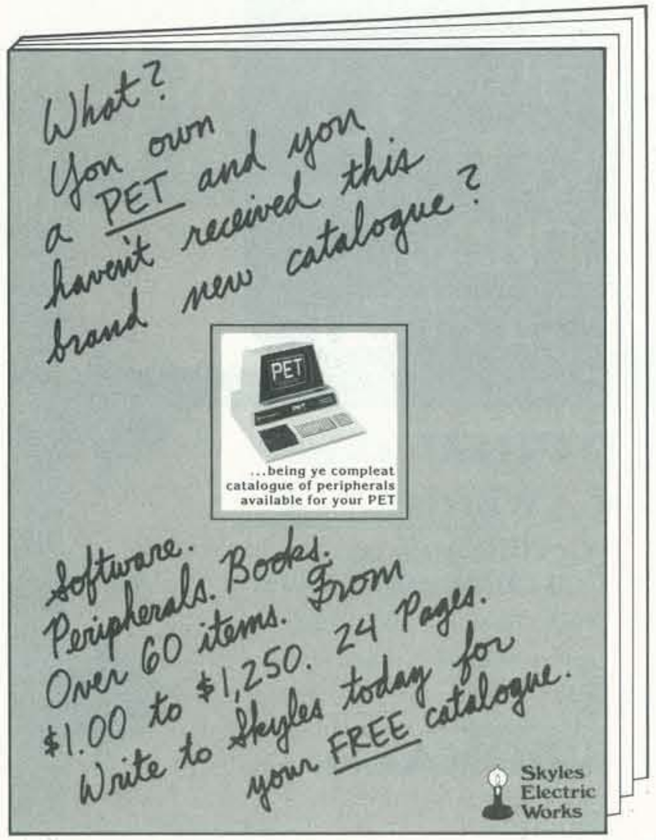
(continued from page 36)

than any technology since the automobile, if it hasn't already. Brand devotes an extensive twelve-page section to computer technology and its role in the '80's, covering everything from books and magazines to software, terminals, small computer languages, access to computer networks, and more.

Because "the microcomputer or personal computer boom is moving so fast," reviewer Dan Dugan suggests that "magazines are the only way to keep track of it." His column offers thirteen reviews of personal computer magazines, including *Recreational Computing* and *Dr. Dobb's Journal*, which the reviewer calls "the best of the 'community-oriented' computer magazines." This section also contains a write-up about ComputerTown, USA!, a Menlo Park, California, project which offers education and access to computers in the community.

An article entitled "System Incoherence," by Mark Le Brun, points out that the computer is a product of our own environment, a sophisticated tool for a new-age ecology: "Computer systems are environments . . . they have a past and a future, functional niches, finite resources and many other properties associated with complicated systems which occur in nature. Yet, few installations take this fact into account as a general policy. As a result the systems are subject to the same sorts of degeneration caused by neglect (or out-and-out rapacity) as their more organic counterparts."

Each version of *The Whole Earth Catalog* has as its motto the phrase "Access to Tools." This issue covers the foremost tool of our modern age with the same comfortable approach it might once have taken to making one's own candles, and it does a fine job. ■



Skyles Electric Works
231 E South Whisman Road
Mountain View, CA 94041

Change of Address??

Name (please print new information)

Address

City State Zip Country

Please advise us of address changes 60 days in advance. Clip this notice and mail, or send us a reasonable facsimile.

Recreational COMPUTING
P.O. Box E
Menlo Park, CA 94025

Attach label here

Recreational Computing MARKETPLACE

Classified ad space available: $60 per vertical inch. Columns are 3¼" wide.

Pacific Discount Software
Z-80/8080 Software. Hundreds of business programs and languages, utilities and games. North Star, Digital Research, Peachtree, Structured Systems, etc., etc. We have it all! We beat all advertised prices on 28/8080 software. Computer Clubs, call. 3204A Production, Oceanside, CA 92054; (714) 439-6201.

PCNET PAN
An electronic mail package, PAN allows PET owners to send and receive messages over the telephone network. Entirely written in BASIC, PAN permits immediate message transmission, or unattended transmission at a specified time. For more information, please send a *stamped, self-addressed, business size envelope* to: PEOPLE'S COMPUTER COMPANY, PCNET Project, P. O. Box E, Menlo Park, CA 94025.

ComputerTown, USA!
A grassroots, economical model of how to offer everyone in a community of 27,000 the opportunity to use a microcomputer. Your interest and help is invited. For more information, please send a *stamped, self-addressed, business size envelope* to: ComputerTown, USA!, People's Computer Company, Box E, Menlo Park, CA 94025.

Advertise with Classifieds!
An ideal opportunity to promote your new product or service is our new Classified Section, Marketplace. Priced low at only $60 per vertical inch, here's your chance to test a hungry hardware/software market. Contact Michael Madaj, Advertising Director, *Recreational Computing*, P. O. Box E, Menlo Park, CA 94025, (415) 323-3111.

ADVERTISERS INDEX

6th COMPUTER FAIRE

Conference & Exposition On
Intelligent Machines for Home, Business & Industry

SAN FRANCISCO CIVIC CENTER APRIL 3-5, 1981

THE BIGGEST
Don't miss the largest microcomputer-related conference and exposition in the United States.

BUSIEST
"...teeming with people, new products, conferences, and good parties..."
— InfoWorld

MOST CAPTIVATING &
See more than 400 exhibits by companies like Radio Shack, Commodore Business Machines, Diablo Systems, Inc., Cromemco, Digital Research, Digital Equipment Corp., Byte Magazine, and many others.

FASCINATING
Hear more than 100 speakers, on as many subjects. This is an idea-sharing conference, and you just might come back as a speaker next year.

MICRO SHOW & CONFERENCE
"...wild mix of exhibitors and attendees ...some of the brightest microcomputer scientists and applications engineers, as well as a horde of personal and business computer buyers...a place to learn of the future as well as profit from the present." — Robert Lively, The Sizzle Sheet

IN THE UNITED STATES
"...the West Coast Computer Faire... draws people from all over the country, and around the world..." — Dr. Adam Osborne "From the Fountainhead"

...AND THE WORLD?
"...Computer Faire appears to be heading for international status, perhaps reaching the size of NCC and WESCON..." — Dr. Adam Osborne, "From the Fountainhead"

MORE THAN 100 CONFERENCE SPEAKERS

Take Your Pick of Conference Sessions For Example:

- Tutorials for Novices
- Tech Talks for Experts
- Low-Cost Business Computing
- Public Information Utilities
- Legal Aspects of Computing
- Inexpensive Educational Computing
- Biomedical Applications
- Computer Graphics & Art
- Computer Music
- Unusual Applications
- Microcomputing for the Physically Impaired
- Social Implications of Computers
- Simulation Applications & Exotic Games
- Users Meetings, e.g.: Apple, TRS-80, Commodore, Forth, etc.

Conference Proceedings Will Be Published & Available at the Computer Faire

MORE THAN 400 EXHIBITS

SAN FRANCISCO
Civic Auditorium and Brooks Hall
(Largest Convention Facility in Northern California)

Friday, April 3rd 9am - 6pm
Saturday, April 4th 9am - 6pm
Sunday, April 5th Noon to 5pm

Registration Includes Conference Program & Exhibits All 3 Days

ON-SITE REGISTRATION: $10

FREE! SILICON GULCH GAZETTE SUBSCRIPTION

Get a free subscription to the Silicon Gulch Gazette. Keep in touch with the microcomputer industry and the 6th Computer Faire. Loaded with news, gossip and tidbits about Silicon Valley (just down the road from the office). Latest information on speakers, exhibitors, special features of the 6th Computer Faire.

COMPUTER FAIRE
333 Swett Road Woodside, CA 94062 (415) 851-7075
Dear Faire Folks: I don't want to miss anything. Please enter my name to receive the **Silicon Gulch Gazette**, for FREE!

Name

Address

City State Zip

CF6