people's computers
PCC
Box E
Menlo Park, CA 94025

Second class postage
paid at Menlo Park,
California 94025 and
additional entry points

93065JOHNSB09 D 2239

BYRON JOHNSON
356 LAGUNA TERR
SIMI VALLEY, CA 93065

Gift Subscriptions!

Send a different gift for the holidays!

Give a gift subscription (or two!) to *People's Computers* this holiday season. For only $8 you can give a gift that keeps coming all year round. Every issue of *People's Computers* will bring exciting and up to the minute articles, listings, interviews, games, announcements and more.

Use the card in the middle of the magazine to order subscriptions for yourself and friends. Send in your gift subscriptions immediately and we'll mail special gift announcement cards in time for the holidays.

$1.50

people's computers

VOL 6 NO 3 NOV – DEC 1977

OUR PET'S FIRST STEPS

OZ GRAPHICS

BIOFEEDBACK & MICROS

SUBMITTING ITEMS FOR PUBLICATION

LABEL everything please, your name, address and the *date*; tapes should also include the program name, language and system.

TYPE text if at all possible, double-spaced, on 8½ x 11 inch white paper.

DRAWINGS should be as clear and neat as possible in black ink on white paper.

LISTINGS are hard to reproduce clearly, so please note:

- Use a new ribbon on plain white paper when making a listing; we prefer roll paper or fan-fold paper.
- Send copies of one or more RUNs of your program, to verify that it runs and to provide a sense of how things work -- and to motivate more of us to read the code. RUNs should illustrate the main purpose and operation of your program as clearly as possible. Bells, whistles and special features should just be described in the documentation unless they're particularly relevant.
- Paper tapes of both the program and runs can provide us with a way to make our own listing if we need to. Then, if you give us permission, we can let CCC (Community Computer Center) sell your program cheaply via paper tape, to further the spread of inexpensive software. Finally, if we are so lucky as to have access to a system on which your program runs, we can try it out ourselves.
- Make sure your code is well documented -- use a separate sheet of paper. Refer to portions of code by line number or label or address please, not by page number. When writing documentation, keep in mind that readers will include beginners and people who may be relatively inexperienced with the language you're using. Helpful documentation/annotation can make your code useful to more people. Documentation should discuss just which cases are covered and which aren't.
- If you send us a program to publish, we reserve the right to annotate it (don't worry, we won't publish it if we don't like it).
- Last but not least, please try to limit the width of your listings: 50-60 characters is ideal. Narrow widths mean less reduction, better readability, and better use of space.

LETTERS are always welcome; we assume its OK to publish them unless you ask us not to. Upon request we will withhold your name from a published letter, but we will not publish correspondence sent to us anonymously. We reserve the right to edit letters for purposes of clarity and brevity.

SUBSCRIPTIONS

U.S. Subscriptions
- ☐ $8/yr. (6 issues)
- ☐ $15/2 yrs. (12 issues)
- ☐ Retaining subscription @ $25 ($17 tax deductible)
- ☐ Sustaining subscription @ $100+ ($92+ tax deductible)

Foreign Surface Mail
- ☐ add $4/yr. for Canada
- ☐ add $5/yr. elsewhere

Foreign AIRMAIL
- ☐ add $8/yr. for Canada
- ☐ add $11/yr. for Europe
- ☐ add $14/yr. elsewhere

Back issues, $1 each; indicate Volume and Issue number, how many copies of each. An order card is at the center of the magazine.
- Vol. 1, No. 3
- Vol. 3, No. 1
- Vol. 4, Nos. 3, 4, 5, 6
- Vol. 5, Nos. 2, 3, 4, 5, 6
- Vol. 6, No. 1

Foreign Distributors of *People's Computers*

Vincent Coen
LP Enterprises
313 Kingston Road
Ilford, IG1 1PJ
Essex, UK

Comicro AG
Baderstrasse 281
CH-8003 Zurich
SWITZERLAND

Pan Atlantic Computer Sys.
Frankfurter Strasse 78
D61 Darmstadt,
WEST GERMANY

Home Computer Club
1070-57 Yamaguchi
Tokorozawa, Saitama, JAPAN

Kougakusha Publ. Col, Ltd
Haneda Biru 403, 5-1
2-Chome, Yoyogi
Shibuya-Ku, Tokyo 151
JAPAN

Computer Age Company, Ltd
Kasumigaseki Building
3-2-5 Kasumigaseki
Chiyoda-Ku, Tokyo 100
JAPAN

ASCII Publishing
305 HI TORIO
5-6-7 Minami Aoyama
Minato-Ku, Tokyo 107
JAPAN

People's Computers is published bimonthly by People's Computer Company, 1263 El Camino Real, Box E, Menlo Park, CA 94025.
People's Computer Company is a tax-exempt, independent, non-profit corporation, and donations are tax-deductible.
Second class postage paid at Menlo Park, California, and additional entry points.

people's computers

VOL 6 NO 3
NOV - DEC 1977

CONTENTS

LETTERS

Hey, PCC People!

I understand your fervor to upgrade the journal, but please don't outgrow me and the other True Beginners in computing. There are plenty of slick, sophisticated publications for people who really understand computers; I know, because my husband gets scads of them. But, I subscribe to you (*twice* – at home and at school), and I fervently hope you won't become incomprehensible to me like all the others.

Also, I hope you continue publishing articles on computers in the schools and tales of other interactions between kids and computers. No other computer magazine, so far as I know, has the kinds of articles you used to have to encourage and guide not-very-knowledgeable adults in leading youngsters to computers. You made me dependent upon Joanne Verplank and now you appear to have exorcised her! (At least tell me how to get in touch with her, please.)

I appreciated Betsy Rosen's bibliography of computer books for children (Vol 6, No 1), but was surprised she omitted Don Spencer's numerous paperbacks (Camelot Press). They are relatively inexpensive, very easy to read, and my schoolkids like them.

Don't desert the kids and us overgrown novices!

Jane McCrohan
102 Council Place
Liverpool, N.Y. 13088

We won't desert you – as I see it, kids and overgrown novices are the biggest potential market around. School stuff is harder to come by over the summer months, but the drought is over – see Harvey Cohen's article on teaching math using OZ graphics. And I'll be reporting on efforts to integrate computers into the day-to-day curricula of a local elementary school. Joanne has not been exorcised, merely taking a breather, and promises to have an article ready for us soon.

Several people have mentioned kids' computer books not included in Betsy Rosen's article – we'd love to have reviews of them. How about sending in a paragraph or two describing them? Not just you Jane, but Bob Kahn and all you others out there!

▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○

Dear People's Computers (or whatever you call yourselves),

I liked the game of EXAGON in the May-June issue and hope you will continue to print stuff like that in forth-coming issues. If I had access to a computer (not till school starts) I would submit stuff myself. Anyway, the reason I'm writing, could someone design and program a Star Trek with the following features:

1. optional two or three dimensions.
2. multiship control option.
3. multiterminal ability option.
4. human control of enemy ships and/or aliens that have a goal and don't just run into each other.
5. communication between terminals.
6. set size and number of 2 or 3 dimensional quadrants.
7. save and restoring capabilities.

and the following list of commands:

1. WARP CHANGE
2. SHORT SCAN*
3. LONG SCAN
4. ENERGY BEAMS
5. PROTON TORPEDOES
6. TRACTOR/REPULSOR BEAMS
7. TRAVEL
8. STATUS REPORT
9. DAMAGE REPORT
10. HIGH WARP CHANGE
11. EMERGENCY WARP CHANGE
12. Energy beams @ target
13. Proton torpedoes @ target
14. tractor/repulsor @ target
15. SHORT RANGE TRACK
16. GALACTIC UPDATE
17. COMMUNICATE WITH OTHER REBELS
18. SAVE game
19. RESTORE game
20. COMPUTER CONTROL:
    A. Dist. and ° to aliens.
    B. auto-move, fire (M5)
21. DAMAGE CONTROL
22. ENERGY ALLOTMENTS

* on the short scan it shouldn't be for just the current quadrant but for an equal number of parsecs around the ship so that if you're near the border between 2 quadrants instead of seeing only the one you're in you see part of the one you're in and part of the one beside you.

Plus anything else anyone can think of.

Tat's all four nouw.
SINcerelee,
Douglas "Dit-Dit" Philips
McCombs Rd RD#2
Box 329
Venetia, PA 15367

P.S° may your compewter never byte U.
P.S.S° U can print this lether. (PLEASE)

Another Star Trek? Whew! Please do submit programs, articles, whatever – see the guidelines on our inside cover.

▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○

I am interested in Computer Assisted Education, especially the PLATO and Multitutor systems which both use the Computer language TUTOR. I would also like to learn about other languages which would be suitable for CAI such as SMALLTALK and PILOT. I would like to correspond with others who are interested in this area.

I also enjoy computer games and believe that they have an important place in CAI. On top of being educational, they are fun and relaxing to play. I have approximately 400 games which are stored on a 2400 ft tape and on listings. I would enjoy trading programs with other people who enjoy computer games.

I wish you good luck and success with your PCC magazine. Long live the dragon!

Sincerely yours,
Bruce A. Carter
7786 South 300 West
Union MIlls, IN 46382

P.S. Science fiction type games are especially desired. Thanx.

We'll continue to publish articles and info on PILOT, as we have in the past. As information on SMALLTALK becomes available we'll keep you posted; for starters, see Dennis Allison's comments on SMALLTALK and other languages on pages 18 - 21. And don't miss Harvey Cohen's article (pages 52 - 58) on how he uses a graphics-oriented language to teach math to kids.

▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○

Dear Phyllis,

Congratulations on your new format! One problem I am having with old issues of the PCC newspaper is that slowly but surely it promoted disintegration of my copies. This was sometimes accelerated by the rough handling the U.S. Post Awful gave them. Yet I still try to preserve them for reference purposes. The new format will help muchly.

As a matter of fact, I was researching on possible sources of money to finance an educational film about home computers when I came across an old letter of mine that was printed in Vol. 5, No. 2, pg 33. Then it occurred to me: your latest issue with its new format and the announcement about California's embrionic Personal Computer Network almost seemed to be an indirect response to that letter! Coincidence? Even if it were, it demonstrates that *People's Computers* does a remarkably good job of keeping in touch with the people.

By the way, you might be interested in knowing that the Personal Computer Network was not the only innovative spin-off from the First West Coast Computer Faire. The Faire was a catalyst which led to the incorporation of ATRA and our first commercial venture: Star Jewels. Advanced Technology Research Associates formed as a group of computer hobbyists – a group that was different because we were also active science fiction fans. For over two years we brainstormed and researched as we explored the latest computer technology. We came upon the idea of selling Star Jewels when we were trying to figure out a way of financing a trip to the Faire for our group. The response of the people at the Faire was fantastic! Now we are planning to expand our sales so we can finance the purchase of group-shared peripherals.

Oh yes, I must not close this little missive without a compliment on those two excellent articles you printed on robot pets. With pet robots and a bit more positive mass media image, such as that presented by C3P0 and R2D2 in Star Wars, the public may yet overcome its fears of these devices. Not since the great Robby the Robot in Forbidden Planet has more been done to show the positive constuctive values of what will soon become man's greatest servants. Once again, keep up the good work!

Frank Stodolka, Treasurer
ATRA, Inc.
Box 456
Minneapolis, MN 55440

Thanks muchly for your kind words. You'll be pleased to know that we expect to see more robots parading across our pages in the near future.

○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○

Dear People's Computers,

I'm a high school computer freak writing from the Los Angeles area and I listen to radio station KMET. Every weekday night, DJ Jim Ladd is on. The DJs at KMET give a lot of their opinions on the air which is generally a good thing but sometimes they don't have all the facts. This is unfortunate, since they influence a lot of public opinion. Anyway, Jim Ladd, although he's pretty cool otherwise, doesn't like computers. He has about the same opinion on computers as a lot of people: as gigantic, impersonal machines that mainly screw up whatever they do with disastrous results. He says, "I don't mind computers when they're used for medical research and things like that, but. . ." and goes on to complain about loss of privacy, etc.

Whenever he announces the number for Commuter Computer, a carpool arranging service, he says, "Why not ask a friend or someone on the street? I don't see why you need the computer. . ." when it's obvious that without the computer you'd have to contact thousands of people to find a good match. Anyway, can you send him something on the benefit of computers? As it is now, he's irritating to me and reinforcing lots of people's prejudices. I've already written to him, but I think you would be more effective.

Write to: Jim Ladd, KMET, 5746 W Sunset Blvd., Los Angeles, CA.

Finally, something for myself: The only computer I have access to runs only FOCAL. Do you have any software in FOCAL? Also, can you send me some info on your publications?

Sincerely,
Joe Boyle
1013 Lake Forest Dr
Claremont, CA 91711

We'll see what we can do – meanwhile readers from the Los Angeles area might want to call Jim Ladd.

○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○

Gentlebeings:

Are you still interested in a cheep computer? My idea is based around a single chip micro. The range is quite restricted, as only the F8/1 and the 66XX offer 2K ROMs, though perhaps the 6801 and Z8 will too. ROM space is cheap and a good language needs all it can get. If they let you play with the microcode, that's even better, though you will need a global optimizing simulator (program) to make a useful gain. RAM would be 4K dynamic, operated jointly by the processor, keyboard, and screen. The keyboard would be a calculator type (Chomerics prices at 1¢/key) hardware with ASCII layout. The screen would be built in (9" tubes are cheapest). A UART tape port completes the set.

It could be quite powerful; an Almost-Pocket-Sized-Computilator. Fully programmable, I hope it sells for under $300. With the built-in keyboard, consider an expanded 256-character set, including characters that enter as one and echo as a string (for, let, next, etc.) and also special graphic characters. This could shorten string-processing space a lot (re: CASUAL, DDJ Nov/Dec 76).

Hoping this finds you,
Andrew Hay, Esq.
12 Holden Wood Rd.
Concord, MA 01742

We sure are interested in a cheep (!) computer, especially one designed to allow beginners to do interesting stuff without knowing much. Keep us posted on your progress!

▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○▲○

PET news and reviews

PET Update

Commodore's PET is a self-contained, factory-assembled unit that contains a 6502 microcomputer, keyboard, CRT display (40 columns, 25 lines), 1000-baud tape cassette, and memory. For $595 you get 4K of user memory (or 8K for $795) plus the 14K needed by an 8K BASIC interpreter, a 4K operating system, a 1K diagnostic routine, and 1K machine language monitor. The PET's expanded 8K BASIC contains strings, integers and multiple dimension arrays. It has high precision (10 significant digits), floating point numbers, and direct memory access through PEEK and POKE.

The system weighs 44 pounds, is 16.5 inches wide, 18.5 inches deep and 14 inches high – about the size of a portable TV but a somewhat more awkward shape to handle. The 73-key calculator-style keyboard includes a calculator-style numeric keypad. The 64 ASCII characters are available without using a shift key; the shift key makes 64 graphic and reverse field characters accessible from the keyboard. The graphic characters can be used to play games, plot, or draw pictures. As delivered, the keyboard provides upper case, but POKE 59468,14 will access lower case instead of 26 graphic characters.

In our last issue, we carried an interview with Chuck Peddle, father of Commodore's PET computer. Here is an addendum to that interview, based on a discussion with Chuck when I picked up one of the first PETs to be delivered in the San Francisco area. Numerous rumors about the PET have been circulating – here's what Chuck had to say as of October 10.

Rumor 1: 90-day delivery committments aren't being met. 'True'; delivery of the 8K units is running at about 100 days and will probably do so through October.

Rumor 2: Production of the 4K system has been cancelled. 'False'; development of a 4K production line has been postponed until the order backlog and production line for the 8K systems are in better shape. Bottlenecks created by parts' shortages should be straightened out by the latter part of October.

Rumor 3: 4K systems won't be in stores until early 1978. 'It's possible.' Individuals who order 4K systems get priority over dealer orders; orders placed through Mr Calculator stores are merged by date of order with mail orders placed directly with Commodore.

Rumor 4: The IEEE bus interface does not come as standard PET equipment. 'False.' The IEEE *interface* is provided but the IEEE *connector* is not – they're expensive (about $50). Commodore will sell a connector cheaper than the IEEE one.

Commodore has not yet announced which large retail store(s) will be carrying the PET. By November 1 Commodore expects to have delivered a minimum of 750 8K systems, and to have cut delivery time back to 30 - 60 days. By early November the 8K PET is also likely to be available in stores.

Commodore hopes to deliver in January its first 200 printers with a price tag around the cost of a 4K PET ($600). (At a recent computer conference, a PET was on display in the booth of Practical Automation, a printer manufacturer, but no official announcement has been made.) The first floppy disk Commodore will offer will cost 'more than a 4K system'.

In mid-September, the American Stock Exchange stopped trading Commodore stock, based on allegations by the Montreal Stock Exchange. Following an investigation, the American Exchange allowed a resumption of trading. Peddle views Commodore as 'cleaner' than companies that have not been investigated; he notes that Commodore is taking steps relative to the Montreal Exchange. Peddle says 'I think both the rumors and the kind of action that occurred with respect to the Montreal Exchange are unfair. We don't know where the unfairness is coming from – whether Commodore has enemies or whether we've done something that frightens people.'

Finally I asked Chuck why the 8K PET costs $200 more than the 4K version – $200 is a lot of money for 4K of memory. I got the 'official' answer, that the larger system requires extensive testing and burning in, but I suspect the real answer is that at this time people are willing to pay $200 to get double the memory.

When *People's Computers'* Editor Phyllis Cole asked:

WHAT ARE THE ADVANTAGES AND DISADVANTAGES OF THE TRS-80 VERSUS THE PET?

Commodore's Chuck Peddle replied:

PET photo courtesy of Utter Chaos

For people who like to put things together, it's an advantage that the TRS-80 is a modular system – Commodore deliberately chose to produce an integrated package. The TRS-80 keyboard is comparable to a typewriter keyboard, which wasn't designed for doing control functions. The difficulty in doing control functions has been overcome by adding special keys to control input/output devices. Commodore has tried to match the multi-key functions available on the more expensive intelligent terminals so as to give more keys in the same space. We'll let the market decide which is better.

The fact that the TRS-80 RAM is expandable up to 16K inside the unit (although I don't know whether such an addition requires an additional power supply) may be seen as an advantage. Commodore allows a 4K PET to be expanded to 8K. The real issue will come in comparing the memory extension boxes.

A UL listing of the PET will be secured by Commodore as soon as full production is reached, as per requirements of the Underwriter's Laboratory.

The software approaches are different on the two systems. The PET has built-in software – such as its editor, extra graphics, and full file system – to make the system easy to use. The PET operating system doesn't depend on a floppy disk for expansion. The PET has more graphics, and the graphics are more easily controlled than those on the TRS-80.

Commodore and Radio Shack see the market in two different ways. A potential buyer needs to read the specifications of a PET and try one out in order to decide whether the significantly larger amount of software available on the PET means something to him as a person. I feel that the claims for the TRS-80 are misleading relative to their ability to store and manipulate data and files. I think that a user should really evaluate his need for data handling before buying any product in the computer field. It's not the number of tapes that make good software. It's what can be put in a program and what the program can do that has to be measured. Sometimes multiple low-cost or free items indicate a problem and not an opportunity.

Radio Shack's Don French replied:

I think that they're totally different type systems. They have a $795 8K system and our 16K system costs only $889 – a $94 difference. They have a different BASIC from ours, and they have a non-standard keyboard. Our extended BASIC, which will be available by the end of the year, will do more than their 14K total system does. It's basically because of the differences in the Z-80 and the 6502. The 6502 has a faster instruction set at the same clock speed, but the Z-80 has a more memory-efficient instruction set, which means that you can provide a more powerful processor in memory size, which means that you can provide the same program in less memory with the Z-80 than you can in the 6502. It's just a trade-off in the type of system you want to design. Commodore chose the 6502 because it's made by a division of their company; Tandy Radio Shack chose the Z-80 just because we thought it was a more powerful unit.

The TRS-80 is listed with the Underwriter's Laboratory and the PET is not, which means that schools can buy our system, and some schools may not be able to buy a PET. You must have a UL listing to sell assembled electronic equipment in some areas, such as Los Angeles, Sacramento, and Oregon. In part we delayed introduction of our system until we acquired our full UL listing.

The TRS-80 is a two-piece system. Ours has a real keyboard on it, which means that with our system you don't have to buy the video screen. Many hobbyists have monitors already, and can use our $399 system directly with their own monitors and cassette recorders.

As for advantages of the PET over the TRS-80, there will be none as soon as we come out with our Level II BASIC. People owning a 4K system can replace chips to update to Level II BASIC at a cost of slightly over $100. The PET does have the IEEE bus, if you classify that as an advantage, but there's a very limited market for that type of bus structure.

Our PET's First Steps

BY PHYLLIS COLE, EDITOR

We have a PET or perhaps I should say a PET has us – at any rate, the relationship is pleasantly symbiotic. The 'we' of this article is not an editorial we, it refers to members of a group of computer professionals – including your editor – who have purchased a PET as part of a project aimed at integrating personal computers into the daily routines in a local school. You'll be kept posted on our activities in the pages of *People's Computers.*

Our group chose a Commodore PET for four reasons:

1) At the time, it was the only announced system that the school could afford.
2) Its integrated package (display, keyboard, and tape unit in a single cabinet) was easier for a child to manage than separate components would be.
3) Its quality BASIC featured strings, arrays, floating point, graphics and linkage to machine language subroutines.
4) It was manufactured locally – a big help, since we knew we would be receiving an early production model.

The specifications of the PET are given in the introduction to 'PET Update' on page 6.

Our PET Stumbles. On Monday, October 4, we picked up PET number 54. When we turned it on we found that 1K of the 8K was taken up by tape buffers, the program stack and other system requirements, leaving only 7167 bytes for us. We soon found two errors caused by inadequate checking during production: a memory chip was flaky and 5 keys on the keyboard weren't making contact. Other 'problems' turned out to be:

a) nobody told us to remove the transparent packing tape from the key tops – when it began to peel, we feared plague had set in.
b) we did not understand certain features and conventions of the system since minimal documentation accompanied early systems.

Our Pet Goes to the Vet. The following day we swapped PET 54 for PET 57: we've been up and running almost constantly since then.

It took about two days to get used to the PET's calculator style keyboard, but get used to it we did. We came to accept the small size of the keys, their closeness, their arrangement, and the way contact with them must be made. In the short time that we've had the machine it has been extremely reliable. Three times in our first 80 hours of use the processor stopped – reputedly due to voltage drops in our power lines. This is the one disadvantage of not having a front panel – we had to power on and off to restart, thus losing the program in memory.

Our PET Starts to RUN. The PET has a number of nice features – we've space to mention just a few. You can get lower case displayed (instead of 26 of the graphic characters) by doing POKE 59468,14. Most versions of BASIC will STOP if you type just a carriage return for INPUT. This annoying behavior, particularly irritating to kids, can be bypassed in PET BASIC by using GET C$, which accepts a single character from the keyboard. Input and output are facilitated by the status word ST, logical device numbers and a WAIT command. Last but by no means least is the fact that the vast majority of error messages are not only comprehensible but even useful!

The on-screen editor that comes with the PET is the nicest one we've seen on a small system. For those unfamiliar with on-screen editors, here's a brief example: suppose you type in a BASIC program, run it, and get an error message which indicates to you that you typed 'Y' where you meant to type 'X'. Assuming that the text is still on the screen, you can move the cursor to the 'Y' then type 'X' and press RETURN: that line will be automatically recompiled. Now move the cursor to the word 'RUN' which you typed before, press RETURN, and the program will be executed once again.

Our PET Speaks. PET BASIC is Microsoft's latest, and is very similar to other microcomputer BASICs. Much available software should be easily converted for use on PETs. The BASIC seems to have very few bugs, and the ones that do exist are not encountered in simple programs. For those of you who want to get a head start on learning it, I recommend Jerry Brown's *Instant BASIC* ($6 from the PCC Bookstore) which deals with an almost identical dialect.

Dancing in the Dark. The temporary documentation provided with early systems contains only a brief summary of commands. By November 1 an instruction tape should be available. We have had to explore the system on our own, with an occasional assist from Commodore.

Files on cassette tapes can be accessed by name, but the file system is not quite so complete as those on more expensive micros such as Processor Tech's SOL. Since minimal software documentation has been distributed, we had to experiment alot before we could read data files reliably. By the way, we've tried out a variety of 'low noise' audio cassette tapes – differing brands, quality, and prices – and had good luck with all of them.

PET's Got Rhythm. With the aid of the PET's built-in timer, we timed most of the instructions on our unit. Based on these results, it is a simple matter to speed up the frequently executed portions of programs. Some of our results are printed in the timing table that accompanies this article. We consider the numbers to be reasonably accurate since they predicted within ±5% the benchmark program timings printed in 'BASIC Timing Comparisons' by Tom Rugg and Phil Feldman in the October 1977 *Kilobaud.* By the way, PET BASIC came in about 4th fastest out of the 30 BASICs examined by Rugg and Feldman.

Jumping for Joy. Was it worth getting an early system despite potential bugs and limited documentation? Yes – otherwise we most likely would have had to delay implementing our school program until September 1978. Besides, we're having great fun! And having a computer at home is a fine way to reduce – we're too busy PETting to eat or sleep. So far we have got running a PILOT interpreter, a number of PILOT programs, a tape file utility, and a drawing program. The drawing program (written by another member of our group) is presented below. There will be more PET software in future issues of *People's Computers.*

DRAWING PICTURES ON THE PET

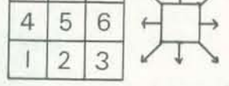
The PET has a set of 64 graphic characters, plus their 64 reverse video (black on white) counterparts. It also has cursor control keys. One might expect to be able to draw a picture simply by running the cursor around and pressing graphic keys. Although possible, it is quite awkward in practice, because the graphic characters and some, but not all, of the cursor controls require the shift key to be held down. It is rather like trying to thread a needle while running the 100-yard dash.

If you have tried to draw on a PET, you will appreciate the program below. It makes it possible for a human being with one or more fingers to draw pictures for entertainment. There is no provision for saving the pictures on tape, although that could be added quite easily.

Type the program, check it for errors and SAVE it on tape before attempting to RUN it. An erroneous POKE could be disastrous.

The Target. The program treats the screen as a grid of cells, 40 across and 25 down. When it starts, it blanks the screen and displays a large round dot character (the 'drawing symbol') in a cell near the center. That cell is the initial 'target' cell. A white square blinks at you occasionally to let you know where the target cell is.

To draw a picture made of dots, use the program's target-motion keys. They are not the same as the PET's cursor keys. Instead, the digit keys 1 through 9 are used to make the target move one cell in any of eight directions.



Pretend the target cell is on the '5' key. To move it left, press '4'; to move it up and right, press '9'; and so on.

Whenever the target moves, it will inscribe the drawing symbol in its new cell. In a few minutes, you should get proficient at drawing a picture with dots.

The Drawing Symbol. When you are tired of dots, press any graphic key on the left side of the keyboard. Shifting is not necessary. The graphic character on the key will become the new drawing symbol. It will be inscribed in the target cell.

Press RVS, and the color of the drawing symbol will be reversed. Subsequently chosen drawing symbols are not affected.

To erase, make SPACE be the drawing symbol. The reverse SPACE draws white stripes. DEL erases the target cell without changing the drawing symbol. You can 'un-DEL' using the '5' key.

TIMING TABLES

BASIC STATEMENTS AND I/O

construct	*approx. time (millisec)*
FRE	1 to 10
PEEK, POKE	1
TI$	3 to 4
TI	1
GET	1 to infinity
POS	1
PRINT X or PRINT	15 to 19
PRINT X$;	14+LEN(X$)/2
READ X and DATA 3	9
REM	0.2 to 2
RESTORE	0.3
TAB	2
SPC(N)	1+0.6*N
FOR I=. . .NEXT I	4.0+(1.6 each)
STEP	1.3
IF	0.4
GOTO or GOSUB	1.1
ON A GOTO or GOSUB $L_1, \ldots, L_M$	0.5+(0.3*A) +(0.2*M)
RETURN	0.9

Using colon, :, saves 0.6 over new line.
SAVE or LOAD
15 sec + (2 sec per 100 char)
i.e. 500 baud

STRING FUNCTIONS

function	*approx. time (millisec)*
+	0.5+(0.2 per char)
ASC	1
CHR$	1.2
LEFT$, RIGHT$	3+(0.025 per char)
LEN	0 to 8
MID$	4+(0.025 per char)
STR$	7 to 10
VAL	1.3
=, <>, <, <=, >, >=	3 to 4

ARITHMETIC FUNCTIONS

function		*approx. time (millisec)*
ABS		0.6
ATN		42
COS		27
EXP		27
INT		1.2
LOG		23
RND	RND(−1)	1.0
	RND(0)	0.9
	RND(1)	4.1
SGN		1.1
SIN		25
TAN		50
user FN		2.4

cont'd

ARITHMETIC OPERATORS

symbol		approx. time (millisec)
↑	0↑B, 1↑B	0.3
	2↑B	32
	else	50 to 100
/	0/B, A/1	0.5
	else	2 to 5
*	0*B, A*0	0.4
	else	1.5 to 3
+		0.3 to 1
–		0.3 to 1
=, <>, <, <=, >, >=		0.7
AND, OR		1.7
NOT		1.4

VARIABLES AND CONSTANTS

item	approx. time (millisec)
A, A$, A=, A$=	0.7 to (0.7+nv*0.1) nv = no. of variables in program
AA, AA$, AA=, AA$=	0.2 more than above
A%	0.3 more than A
A%=	0.6 more than A=
999	1 per digit
.999	0.7+(4.2 per digit)
E16	0.2+(0.4*exponent)
E-16	0.2+(3.0*exponent)
"ABCDE"	(0.6 to 0.7)+(0.02 per char)
M(I,J, . . .)	(1 to 1.5)* (no. of subscripts)

TIMING PROGRAM

```
100  N=300
200  T1=TI
300  FOR I=1 TO N
400  REM PUT TEST CONSTRUCT HERE
500  NEXT I
600  T2=TI
700  FOR I=1 TO N
800  NEXT I
900  T3=TI
1000 PRINT 1000*((2*T2-T1-T3)/(60*N))
1100 END
```

MEMORY USAGE (IN BYTES)

BASIC 1028 (I/O buffers, tables, etc)
each statement
- 4 for line number and following space, regardless of the line number
- 1 for each BASIC keyword
- 1 for each other character, including RETURN

each variable with a value assigned, regardless of spelling or value takes 7 bytes; for string variables, add the length of the string
each array (N.B., size includes 0th element) take f*(size+1)+(2 per dimension) where f=5 for floating point arrays, f=2 for integer arrays, and f=3 for string arrays.
The system slows down noticeably when memory is nearly full.

Other Features. You can draw with the graphics on keys 1 - 9 by SHIFTing. These are the only keys in the program affected by SHIFT.

To move the target without changing the picture, get rid of the drawing symbol by pressing either of the CRSR keys, then use the digit keys to move the target. When you are ready to draw again, press a graphics key or RVS.

When you want to admire your drawings without the screen blinking at you periodically, press RETURN. Then, to make the target cell blink again, type any other key; '5' is a good choice.

To start a new picture, press CLR. To stop drawing so you can do something else with your PET, first press RETURN and then press STOP.

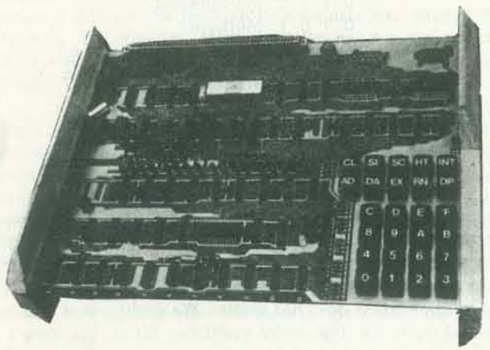
Have fun! □

```
                    1 REM PET DRAWING PROGRAM
                    2 REM COPYRIGHT 1977 "PEOPLE'S COMPUTERS"
                    3 REM PERMISSION TO USE, NOT TO SELL
Initializes constants.
                   10 P1=59409: P2=52: P3=60
Sets timing for blinking.
                   20 WH=50: BL=53: WT=5
Puts 'SHIFT-Q' center screen.
                   30 Y=12: X=20: GR=ASC("  ") AND 127
Cleans screen.
                   40 PRINT "  "
                   50 GOTO 2300
Looks for keystroke.
                  100 GET C$: IF C$="" GOTO 3000
Keystroke to unshifted ascii.
                  150 C=ASC (C$) AND 127
Checks target blinked off.
                  200 IF FL>=WH THEN GOSUB 4000
RETURN causes long blink.
                  250 IF C=13 THEN FL=-1.0E8: GOTO 100
Reset short blink.
                  300 FL=WH-WT
Handles number key.
                  400 IF C$>="1" AND C$<="9" THEN 1700
Handles graphic key.
                  450 IF C>=32 GOTO 1000
DEL blanks target cell.
                  500 IF C=20 THEN POKE L,32
Handles RVS.
                  600 IF C=18 GOTO 1200
Handles CRSR.
                  700 IF C=17 OR C=29 THEN GR=-1
CLR-HOME causes restart.
                  800 IF C=19 GOTO 30
                  900 GOTO 100
Encode symbol for display.
                 1000 GR=64 OR C
                 1100 GOTO 1300
Reset drawing symbol.
                 1200 IF GR<0 THEN GR=PEEK(L)
Reverse drawing symbol.
                 1250 GR=(128+GR) AND 255
Lines 1300–1500 display the
symbol in the target cell.
                 1300 POKE P1, P2
                 1400 POKE L, GR
                 1500 POKE P1, P3
                 1600 GOTO 100
Lines 1700–2200 move the
target in the direction indi-
cated by the number key.
                 1700 X=X+C+1-3* INT((C+2)/3)
                 1800 IF X<0 THEN X=0
                 1900 IF X>39 THEN X=39
                 2000 Y=Y+1-INT ((C-49)/3)
                 2100 IF Y<0 THEN Y=0
                 2200 IF Y>24 THEN Y=24
Locates new target.
                 2300 L=32768+40*Y+X
Checks for drawing symbol.
                 2400 IF GR>=0 GOTO 1300
Lines 2500–2700 blink the
target if there's no
symbol to draw.
                 2500 GOSUB 4000
                 2600 FOR DL=1 TO WT: NEXT DL
                 2700 GOSUB 4000
                 2800 GOTO 100
Lines 3000–3020 blink the
target while you're not
doing anything.
                 3000 FL=FL+1
                 3010 IF FL=WH THEN GOSUB 4000
                 3020 IF FL=BL THEN FL=0: GOSUB 4000
                 3030 GOTO 100
Lines 4000–4020 reverse the
color of the target cell.
                 4000 POKE P1, P2
                 4010 POKE L, (PEEK(L)+128) AND 255
                 4020 POKE P1, P3
                 4030 RETURN
```


THE DATA HANDLER USERS MANUAL:

PART 6

BY DON INMAN



The DATA HANDLER is a complete microcomputer system on a single PC board based on the MOS technology 6502 microprocessor. The DATA HANDLER can operate at very high speeds as a stand alone microcomputer or dedicated controller for even such high speed devices as disk peripherals. External TTYs or terminals are not needed since the DATA HANDLER contains 26 keyboard switches for full function hardware front control; personal expandability of the system is achieved by using the Altair/IMSAI peripherals on the DATA HANDLER PC board. The DATA HANDLER Bare Bones Kit which includes the DATA HANDLER PC board, PC board stand, 26 keyboard switches, and a complete documentation package is being offered at a price of $89.95. The complete kit is priced at $179.95. This includes the DATA HANDLER PC board, PC board stand, 26 keyboard switches, the complete set of IC's, 1 6502 MOS Technology microprocessor, sockets, LED's, resistors, capacitors, 500 ns memory, and a complete documentation package.

Don Inman is a former teacher, now editor of Calculators/Computers, *who's been working with teachers in the San Jose School District. Under Don's guidance, the teachers have built Data Handlers, complete microcomputer systems based on the 6502 microprocessor, and are now learning to use them. This is the sixth in a series of articles aimed at teaching relatively inexperienced people how to do assembly language programming for the 6502.*

This user's manual is designed to serve both as a self-teaching guide and as an outline for a course at the beginning level of computer science. While it deals specifically with the Data Handler, it can easily be adapted to other microcomputers using the MOS Technology 6502, such as the PET.

The first semester course consists of nine two-hour class sessions, the first two of which were spent constructing the systems. Part 1 of our series covered session 3 of the course: system specifications, binary and hexadecimal notation, and how to do a preliminary checkout of the system. Part 2 of our series covered data transfer, and the use of a simple data transfer program. Part 3 covered session 5: the arithmetic logic unit. Part 4 indexed addressing and Part 5 discussed how to write programs. This article will cover specific methods of programming for multiplication and division.

SESSION VIII – MULTIPLICATION AND DIVISION

The 6502 instruction set has no provision for multiplication or division. However, these operations may be achieved by programming appropriate instructions that do exist. This session will be devoted to developing such programs.

MULTIPLICATION BY SUCCESSIVE ADDITION

```
LABEL   OPERATOR   OPERAND    COMMENTS

        LDX        $( )       Load the first number in X register
GO      LDA        $FC50      Load the accumulator with 2nd number
        CLD                   Clear decimal mode
        CLC                   Clear carry
        ADC        $FC61      Add accumulator to partial sum
        STA        $FC61      Store back as new partial sum
        BCS        HIORD      Branch if carry to increment hi-order
COUNT   DEX                   Decrement the X register
        BNE        GO         Test X register, branch back if not=0
        JUMP       FINIS      Jump if X register=0 to finish
HIORD   LDA        $FC60      Load accumulator with high order
        CLC                   Clear any carry
        ADC        $01        Add one
        STA        $FC60      Store back the result
        JMP        COUNT      Jump back to decrement X register
FINIS   JMP        FINIS      Loop until program is halted
```

MULTIPLICATION

The easiest method to achieve multiplication is by successive addition. One number is used as a counter to tell the arithmetic unit how many times the other number should be added to itself. From the computer's viewpoint this method is very slow, but it works. Let's assume we wish to multiply two 8-bit numbers and need 16 bits to express our result.

As an example, the X-register could be loaded with one number and used as a counter. The second number could be stored in some memory location. We would load the accumulator from that memory, decrement the X-register, add the number in memory, decrement the X-register (testing the X-register to see if we are finished), and continue branching back to add the number from memory and decrement the X-register until it has counted down to zero.

Above is a listing for the symbolic code which might be used for such a program. As an exercise, write the machine language program which would implement this code. The program on page 44 is provided as a sample answer.

Memory location FC50 is set up to hold the multiplicand. The X-register is to be loaded with the multiplier. Location FC61 holds the low order partial product, and FC60 holds the high order partial product. The $ symbol indicates a hexadecimal number is used.

Let's now look at an alternative method for multiplication. Consider a binary multiplication of 6 X 5. Writing it in the form normally used for hand calculation:

```
  0 1 1 0   multiplicand
  0 1 0 1   multiplier
  -------
  0 1 1 0   1st partial product is 6
0 0 0 0     2nd partial product is 0
  0 1 1 0   3rd partial product is 6
0 0 0 0     4th partial product is 0
-----------
0 0 1 1 1 1 0   Result = 16+8+4+2 = 30
```

We notice that if a specific multiplier bit is zero, we get zero for that partial product. If the specific multiplier bit is one, we get the multiplicand for that partial product. We also notice that each partial product is *shifted left* one place as ascending place values of the multiplier are encountered. The order of the multiplication does not matter. We might just as well have worked from the high order multiplier bit to the lowest. In fact, this will work best for computer implementation. The hand calculation would look like this:

```
        0 1 1 0
        0 1 0 1
      -------
0 0 0 0
  0 1 1 0
    0 0 0 0
      0 1 1 0
-------------
0 0 1 1 1 1 0 = 30
```

With this in mind consider the above program.

This program multiplies two 8-bit numbers and produces a 16-bit result. Two memory locations are needed to hold the result. HIORD names the memory location to hold the high order 8 bits, and PARTL names the location where the low order 8 bits will be held. Both are initialized to zero before running the program. Memory locations are also needed to hold the multiplier and the multiplicand. These are named MULTP and MULTC, respectively, in the program.

MULTIPLICATION BY SHIFTING

```
LABEL   OPERATOR   OPERAND    COMMENTS

        LDX        $08        Load X register with counter for 8-bits
        CLD                   Clear decimal mode
        CLC                   Clear carry
LOOP1   ASL        HIORD      Shift hi-order partial product
        ASL        PARTL      Shift lo-order partial product
        BCS        DBL        Branch if carry to increment hi-order
LOOP2   ASL        MULTP      Shift multiplier left into carry
        BCS        MULT       Branch if carry to add partial product
CONT    DEX                   Decrement the X register
        BNE        LOOP1      Branch back if X register not = 0
        JUMP       END        Jump to end if X register = 0
MULT    CLC                   Clear carry
        LDA        PARTL      Load accumulator with partial product (lo)
        ADC        MULTC      Add the multiplicand
        STA        PARTL      Store back in partial product (lo)
        JMP        CONT       Jump back to decrement X register
DBL     INC        HIORD      Increment hi-order partial product
        CLC                   Clear carry
        JMP        LOOP2      Jump back
END     JMP        END        Loop here until halted
```

Each time we pass through Loop 1 the partial products are increased or left alone (depending on whether a carry is produced when the multiplier is shifted left). The partial products are also shifted left to reflect the descending place values of the multiplier. The high order partial product is incremented each time a carry occurs from the low order partial product. The X-register is decremented with each pass through the loop. When it reaches zero, all 8 bits of the multiplier have been tested and we exit from the loop to the end of the program.

Write a machine language program to implement the mnemonic code of this program. De-bug your program and run it. If you get stuck, one implementation is shown on the following page.

DIVISION

The division operation can be achieved by successive subtraction of the divisor from the dividend in a manner similar to that used for multiplication by successive addition. An alternative method will be demonstrated here using an 8-bit divisor and a 16-bit dividend. As can be seen, we examine the dividend from the most significant digit downward until the divisor is less than, or equal to, that part of the dividend. At this point, a 1 appears in the quotient and the divisor is subtracted from that part. When the divisor is larger than the examined portion of the dividend, a zero is entered in the quotient. Our computer can implement this by shifting the dividend left into a new memory location and comparing this number to the divisor. If the divisor is smaller or equal, a one is placed in the quotient. The quotient is then shifted left to make room for the next try.

06AC ÷ 0C

```
                          1 0 0 0  1 1 1 0 = 8E
0 0 0 0  1 1 0 0 ) 0 0 0 0  0 1 1 0  1 0 1 0  1 1 0 0
                    1 1 0 0
                          1 0 1 0  1
                          1 1 0 0
                          1 0 0 1  1
                          1 1 0 0
                            1 1 0  0
                            1 1 0
                              1 0  0
                              1 1  0
                                1  0 0
                                1  1 0 0
                          R = 4 =    1 0 0
```

A flowchart of a typical program and a listing of the program appears on the following pages. The program uses an 8-bit divisor in memory location FD03, a 16-bit quotient in locations FD01 (most significant 8 bits) and FD02 (least significant 8 bits). The quotient appears in FD04. FD00 is used to store the remainder, and the X-register is used for counting.

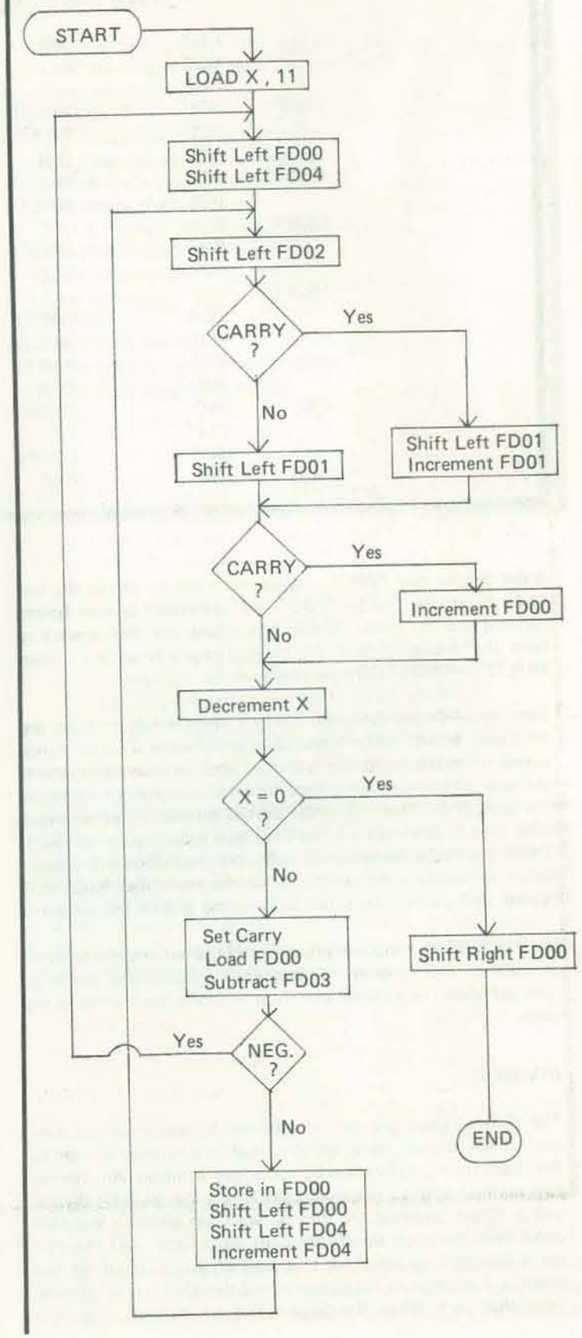
PROGRAM FOR MULTIPLICATION BY SHIFTING

LABEL	LOC	INST	COMMENTS
	FC00	A2	Load X register
	FC01	08	with 8.
	FC02	D8	CLD
	FC03	18	CLC
LOOP1	FC04	0E	ASL HIORD
	FC05	03	
	FC06	FD	
	FC07	0E	ASL PARTL
	FC08	02	
	FC09	FD	
	FC0A	B0	BCS DBL
	FC0B	18	
LOOP2	FC0C	0E	ASL MULTP
	FC0D	01	
	FC0E	FD	
	FC0F	B0	BCS MULT
	FC10	06	
COUNT	FC11	CA	DEX
	FC12	D0	BNE LOOP1
	FC13	F0	
	FC14	4C	JMP END
	FC15	2A	
	FC16	FC	
MULT	FC17	18	CLC
	FC18	AD	LDA PARTL
	FC19	02	
	FC1A	FD	
	FC1B	6D	ADC MULTC
	FC1C	00	
	FC1D	FD	
	FC1E	8D	STA PARTL
	FC1F	02	
	FC20	FD	
	FC21	4C	JMP COUNT
	FC22	11	
	FC23	FC	
DBL	FC24	EE	INC HIORD
	FC25	03	
	FC26	FD	
	FC27	4C	JMP LOOP2
	FC28	0C	
	FC29	FC	
END	FC2A	4C	JMP END
	FC2B	2A	
	FC2C	FC	

Load: FD00 with multiplicand, FD01 with multiplier, FD02 with 00, and FD03 with 00.
Answer: High order in FD03, low order in FD02.

DIVISION

8-bit divisor, 16-bit dividend, 8-bit quotient



DIVISION PROGRAM

16-BIT DIVIDEND;
8-BIT DIVISOR, QUOTIENT, and REMAINDER

LABEL	LOC	CODE	MNEM	COMMENTS
	FC00	D8	CLD	Clear decimal mode.
	FC01	A2	LDX	Load the X register with
	FC02	11		count.
LOOP1	FC03	0E	ASL	Shift left remainder.
	FC04	00		
	FC05	FD		
	FC06	0E	ASL	Shift left quotient.
	FC07	04		
	FC08	FD		
LOOP2	FC09	0E	ASL	Shift left lo-dividend.
	FC0A	02		
	FC0B	FD		
	FC0C	B0	BCS	Branch if carry to
	FC0D	20		HIORD.
	FC0E	0E	ASL	Shift left hi-dividend.
	FC0F	01		
	FC10	FD		
LOOP3	FC11	B0	BCS	Branch if carry to INCR.
	FC12	24		
LOOP4	FC13	CA	DEX	Decrement counter.
	FC14	F0	BEQ	Branch if it = 0 to FINIS.
	FC15	27		
	FC16	38	SEC	Set carry for subtraction.
	FC17	AD	LDA	Load hi-dividend thru
	FC18	00		remainder.
	FC19	FD		
	FC1A	ED	SBC	Subtract divisor.
	FC1B	03		
	FC1C	FD		
	FC1D	30	BMI	Branch if neg. to LOOP1.
	FC1E	E4		
	FC1F	8D	STA	STORE remainder.
	FC20	00		
	FC21	FD		
	FC22	0E	ASL	Shift left remainder.
	FC23	00		
	FC24	FD		
	FC25	0E	ASL	Shift left quotient.
	FC26	04		
	FC27	FD		
	FC28	EE	INC	Increment quotient.
	FC29	04		
	FC2A	FD		
	FC2B	4C	JMP	Jump back to LOOP2.
	FC2C	09		
	FC2D	FC		

LABEL	LOC	CODE	MNEM	COMMENTS
HIORD	FC2E	0E	ASL	Shift left hi-dividend.
	FC2F	01		
	FC30	FD		
	FC31	EE	INC	Increment hi-dividend.
	FC32	01		
	FC33	FD		
	FC34	4C	JMP	Jump back to LOOP3.
	FC35	11		
	FC36	FC		
INCR	FC37	EE	INCR	Increment remainder.
	FC38	00		
	FC39	FD		
	4C3A	4C	JMP	Jump back to LOOP4.
	4C3B	13		
	FC3C	FC		
FINIS	FC3D	43	LSR	Shift right remainder.
	FC3E	00		
	FC3F	FD		
	FC40	AD	LDA	Load the quotient.
	FC41	04		
	FC42	FD		
	FC43	8D	STA	Output quotient.
	FC44	FE		
	FC45	7F		
	FC46	AD	LDA	Wait in this loop for sig-
	FC47	FF		nal (strobe from input).
	FC48	7F		
	FC49	F0	BEQ	Go back if no signal.
	FC4A	FB		
	FC4B	AD	LDA	If signal load remainder.
	FC4C	00		
	FC4D	FD		
	FC4E	8D	STA	Now, output remainder.
	FC4F	FE		
	FC50	7F		
	FC51	4C	JMP	It's all over now!!!!
	FC52	51		
	FC53	FC		

Load:		
	FD00	00
	FC01	hi-order dividend bits
	FD02	lo-order dividend bits
	FD03	divisor
	FD04	00
	FFFC	00
	FFFD	FC

Strobe zero into input port

If you don't have your input and output ports wired yet, the quotient is in location FD04, remainder in FD00.

The next and final article in this series will discuss simple and inexpensive output devices. □

TINY LANGUAGES

SECTION 1: DRAGONSTUFF
BY BOB ALBRECHT
THE DRAGON

We expect to obtain grants and hardware as prizes to further the Tiny Language extravaganza. For starters, our good friend Anonymous has donated $1000 to promote the development of Tiny Languages! We expect to have the contest aspects of the project spelled out in our Jan.-Feb. issue. Meanwhile, let's hear from you!

Last time we said:
It's Tiny Language time again! This time, Dennis Allison and I and (we hope) lots of you people out there, want to design a language that

- Is good for Tiny BASIC type problems and is also good for Tiny PILOT type problems.
- Is designed to be most useful to elementary school kids (at home or at school) and also useful to teachers and parents of elementary school kids.
- Can be implemented in about 4K bytes of ROM with extensions possible in RAM.
- Runs on a Personal Computer whose advanced chip technology controls a color TV with simple graphics.

So, with your help, we want to design a Tiny Language for the home/school computer. We would like this language to be useful for educational and recreational applications. It has to be *learnable* and *teachable* and *easy-to-use*.

OK – I'LL NOW STICK MY NECK OUT:

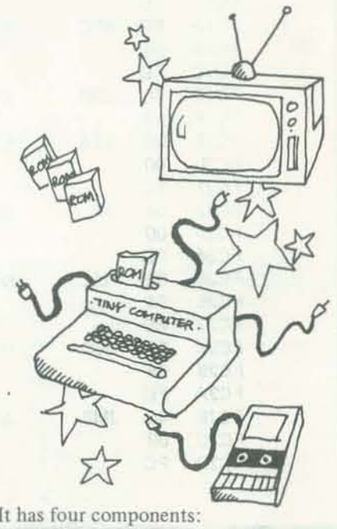
- The most widely used computer language in elementary and secondary school educations is BASIC. There is not even a close runner-up.
- Contrary to conventional wisdom, BASIC is *not* easy-to-learn, nor is it really easy-to-use. Unless, of course, you are one of those gifted people for whom most any computer language is easy-to-learn and easy-to-use. As we move down to earlier grade levels, we find it increasingly difficult to teach BASIC to students or to teachers.
- Enter PILOT. PILOT is easier-to-learn than BASIC, especially for younger kids and teachers of younger kids. PILOT is also easier-to-use (than BASIC) for many educational applications, particularly tutorial dialog. However, for many other educational applications, it is dreadful.
- I am moderately 'fluent' in FORTRAN and BASIC. I began working with elementary and secondary school students in 1962, using FORTRAN. In 1965, with a prolonged sigh of relief, I abandoned FORTRAN, switched to BASIC, and formed SHAFT (Society to Help Abolish FORTRAN Teaching). Now I'm ready for something better than BASIC.
- Unfortunately, I know little about languages such as LISP, SNOBOL, LOGO and SMALL TALK. If you have experience in teaching or learning or using these languages, please share!
- APL makes my teeth rattle.
- I don't like to program, but I do like to use computers – and I like to help students and teachers learn how to use computers. Having to write programs always gets between me and what I want to do with computers.
- In working (well, playing) with elementary school children, I like these computer applications:

1. Games.
2. Simulations.
3. Math recreations (e.g., simple number theory).
4. Math problem-solving.
5. Compassionate drill and practice.
6. Interactive story telling.
7. Graphics and music.

- For some of the above, PILOT is better than BASIC. For some of the above, BASIC is better than PILOT. For some, neither language is suitable.
- So . . . for my purposes, a single language that combines the utilities of PILOT and BASIC would be nice. But, for *me* to use it, it's got to be easy-to-learn, easy-to-use.

HARDWARE

Our Tiny Language might run on a Home/School computer that looks like this:



It has four components:

1. The computer.
2. A TV.
3. Cassette recorder (or floppy disk).
4. A plug-in ROM [Read Only Memory, 4096 to 8192 bytes] containing our Tiny Language.

STRIKE AGAIN

PART II

ARITHMETIC

What kind of arithmetic should our Tiny Language have? Look at some elementary school math books. I recommend books in the widely-used series *Mathematics Around Us*, published by Scott, Foresman and Company in Glenview, Illinois.

GRADES 1 TO 3
Kids learn to compute with whole numbers (non-negative integers) – very little use of mixed decimal numbers.

GRADE 4
- Some addition and subtraction of numbers with one or two decimal places.
- Division of whole numbers, with remainder.
- Very brief introduction to addition and subtraction of fractions.

GRADE 5
- Multiplication of mixed numbers with up to three decimal places.
- Multiplication of fractions.

GRADE 6
- Division of mixed decimal numbers.
- Division of fractions.
- Negative numbers – first time!

So, in grades 1 to 6, we work primarily in the realm of whole numbers. However, keep in mind that, beginning at about 4th grade, we will see a lot of calculators being used by kids at school and at home.

I suggest that, in our initial design of a Tiny Language, we use integer arithmetic. We should allow at least eight decimal digits – but big numbers are fun, so maybe we should think about letting the user increase the limit to more digits.

For example, with eight digits, we soon bog down in exploring the following patterns.

9 x 9 = 81
99 x 99 = 9801
999 x 999 = 998001
9999 x 9999 = 99980001
99999 x 99999 = OVERFLOW

This pattern continues. Be nice to pursue it a little further.

11 x 11 = 121
111 x 111 = 12321
1111 x 1111 = 1234321
11111 x 11111 = OVERFLOW

This pattern breaks down. We would like to get to the breakdown point, see it happen, then talk about it!

For a second type of arithmetic, I would suggest using calculator arithmetic – the type used in inexpensive, four function calculators. Calculator people call it *floating point*, but it is *not* what computer people have called floating point. That is, it is *not* scientific notation. It simple means that the point 'floats' back and forth in the display. Since lots of kids will be using this type of calculator, they will be accustomed to this type of arithmetic.

PARTING THOUGHTS

We're looking for

PARTICIPATION
COOPERATION
ENTHUSIASM . . .

YOUR ideas on how to design a language for kids. We especially want to hear from people who WATCH LOTS OF KIDS LEARNING HOW TO PROGRAM IN VARIOUS LANGUAGES (All kinds of kids – *not* just carefully selected 10 year old geniuses).

– TO BE CONTINUED –

RECOMMENDED READING

1. *People's Computers** PILOT articles – especially see Volume 5, Numbers 4 - 6 and Volume 6, Numbers 1 and 2.
2. 'Tilting at Windmills, or What's Wrong with BASIC?' by Marc Le Brun in *People's Computer Company**, Volume 1, Number 2 (December 1972) – good luck – this issue is out of print!
3. 'Why I HATE My Computer When It Speaks in BASIC' by James W. Garson, *People's Computer Company**, Volume 5, Number 5 (March-April 1977).
4. ' A Critical Look at BASIC' by Dennis Allison in *Dr. Dobb's Journal*, Volume 1, Number 2 (February 1976).
5. 'Playing Games at the Center' by Joanne Koltnow Verplank in *Calculators/Computers* Magazine, May 1977.
6. 'It's Fun/It's Educational – Classroom Computer Games' by Joanne Koltnow Verplank in *Calculators/Computers* Magazine, October 1977.

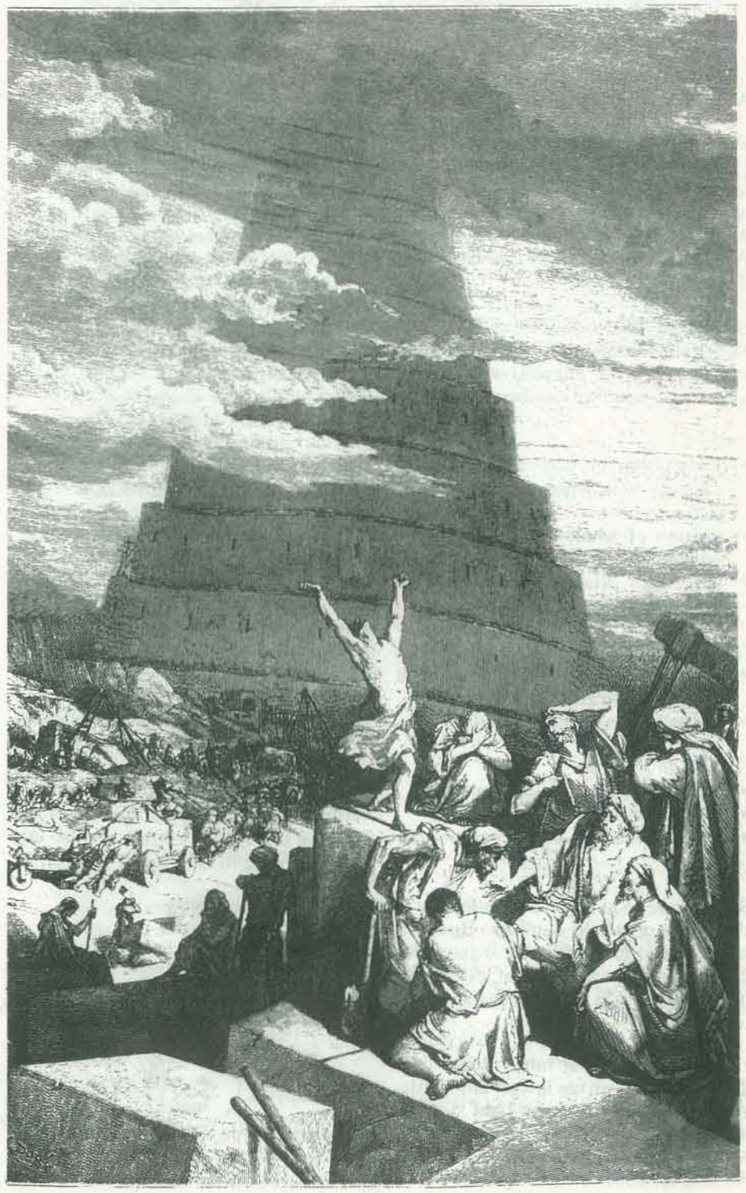
* Note: prior to Volume 5, Number 6 (May-June 1977), *People's Computers* was in newspaper format and called *People's Computer Company*.

MORE

Tiny Languages

SECTION 2: A SHORT GUIDED TOUR OF THE TOWER OF BABEL

BY DENNIS ALLISON



Sometimes it seems there are nearly as many programming languages as there are programmers. Certainly everyone wants to design languages, everyone knows just what is wrong with *the other languages* and how to make his the best.

While lots of people have started to design languages, very few have ever completed the job, implemented a compiler, and convinced people that their language is useful and better. Even so, there are just bundles and bundles of languages in use today.

One of the best ways to learn about languages is to look at the ideas other people have used. There is an enormous literature on programming language design, but most of it is not in general circulation. If you have access to a good computer science library, look in old issues of *SIGPLAN Notices*, a publication of the *S*pecial *I*nterest *G*roup in *P*rogramming *L*anguages of the *A*ssociation for Computing Machinery. The *ALGOL Bulletin*, particularly the early issues, are also fascinating reading. Occasional articles in the other professional computer journals (*Communications of the ACM, Computer Journal, BIT, Software Practice and Experience*, etc.) also discuss aspects of programming language design.

For several years, Jean Sammet has published an annual roster of programming languages. Her book, *Programming Languages* (Prentice-Hall, 1969), provides a survey of 120 languages in some detail but is hopelessly out of date. It is, however, good for starters. The latest copy of her roster appeared in the December 1976 issue of *Communications of the ACM*. The listing below excerpts and slightly updates that listing with the inclusion of Smalltalk.

ALPHABETICAL LISTING OF LANGUAGES

Extracted from a list by Jean Sammet; SMALLTALK material added by Dennis Allison.

ALGOL 60 (*ALGO*rithmic *L*anguage)
A language developed jointly in the U.S. and Europe. Suitable for expressing solutions to problems requiring numeric computation and some logical processes. Has no officially defined input/output. Revised ALGOL 60 (with input/output specifications added) has been approved as an international standard.
Implemented on: Many computers.
Ref: Naur, P. (Ed.), "Revised Report on the Algorithmic Language ALGOL 60," *Comm. ACM*, Vol. 6, No. 1 (Jan. 1963); "Collected Algorithms from the *Comm. ACM.*" (1966 ff.); *ALGOL Bulletin*; International Standard ISO/R 1538-19722, *Programming Language ALGOL.*
Contact: John E. L. Peck, Chairman IFIP/WG 2.1, Computer Science, Univ. British Columbia, Vancouver, B.C. V6T 1W5, Canada; (For the standard) Director of Standards, CBEMA, 1828 L St. N.W., Washington, DC 20036.
Applications Area: Numerical scientific.

ALGOL 68
A very powerful language which includes extensible language facilities and employs a new grammatical method of language definition. Developed under IFIP auspices by Working Group 2.1 (ALGOL). *Not* a compatible extension of ALGOL 60 although it retains the atmosphere of the earlier languages. ALGOL 68 was originally defined in 1968; it has recently been revised by WG 2.1 in the light of implementation experience.
Implemented on: CDC Cyber, IBM 360/370, ICL 1900 series, EL X8, TR4, and DEC PDP-11.
Ref: Van Wijngaarden, A., et al., "Revised Report on the Algorithmic Language ALGOL 68," *Acta Informatica* Fasc. 1-3. 1975. Springer-Verlag, Berlin; Tanenbaum, A., "A Tutorial on ALGOL 68," *ACM Computing Surveys* (to be published in June 1976); Branquart, P., et al., "The Composition of Semantics in ALGOL 68," *Comm. ACM*, Vol. 14, No. 11 (Nov. 1971).
Contact: IFIP/WG 2.1 Subcommittee on ALGOL 68 Support, c/o Robert C. Uzgalis, Computer Science Dept., School of Engineering and Applied Science, UCLA, Los Angeles, CA 90024.
Application Area: Multipurpose.

APL\360
A powerful language with simple syntax and an unusual character set which operates directly on numeric or literal arrays. Most suitable for mathematical problems but has been used in other areas. Based on Iverson's language.
Implemented on: Interactive versions implemented on several computers.
Ref: APL\360-OS and APL\360-DOS General Information Manual, GH20-0850 and *APL\360-OS and APL\360-DOS Users Manual,* GH20-0906, IBM Data Processing Division, White Plains, N.Y. 10504; *APL Quote-Quad*, ACM STAPL/SIGPLAN.
Contact: ————————
Applications Area: Numerical scientific.

BASIC (*B*eginner's *A*ll purpose *S*ymbolic *I*nstruction *C*ode)
A very simple language for use in solving numerical problems, but with some advanced features (e.g. matrix manipulation and string handling statements) in some versions. Developed in both on-line and batch versions. An ANSI standard "Minimal BASIC" is in final approval stage, and work is under way on extensions, to be published later.
Implemented on: Almost all computers.
Ref: BSR X3.60, Programming Language Minimal BASIC (draft standard).
Contact: (For the standard) Director of Standards, CBEMA, 1828 L St. N.W., Washington, DC 20036.
Application Area: Numerical scientific.

BCPL (*B*ootstrap *C*ombined *P*rogramming *L*anguage)
A simple recursive langauge for compiler writing and systems programming.
Implemented on: Many (including British) computers.
Ref: Richards, M., *BCPL Programming Manual*, Computer Lab., Corn Exchange St., Cambridge, England; Richards, M., "BCPL: A Tool for Compiler Writing and System Programming," *Proc. AFIPS SJCC*, Vol. 34 (1969).
Contact: Author; in U.S., Arthur Evans, Bolt, Bernaek & Newman, Inc., 50 Moulton Street, Cambridge, MA 02138.
Application Area: Systems programming.

C
Based on an earlier language, B, which in turn was based on BCPL. Designed for nonnumeric and systems programming.
Implemented on: Honeywell 6000, IBM System/370, DEC PDP-11.
Ref: Ritchie, D.M., Kernighan, B.W., and Lesk, M.E., *The C Programming Language*, Computing Science Technical Report No. 31, Bell Telephone Laboratories, Murray Hill, NJ 07974 (Oct. 1975).
Contact: Authors.
Application Area: Systems programming.

COBOL (*CO*mmon *B*usiness *O*riented *L*anguage)
An English-like language suitable for business data processing problems. Developed and maintained by a committee of representatives from manufacturers and users. A revised ANSI standard was published in 1974, extending and revising the 1968 standard. The 1974 text has also been adopted as ISO Standard 1989. Further development of the language (not the standard) is done by CODA-SYL Programming Language Committee.
Implemented on: Most computers.
Ref: American National Standard COBOL, ANS X3.23-1974; *CODA-SYL COBOL Journal of Development 1973*, Technical Services Branch, Department of Supply and Services, 5th floor, 88 Metcalfe St., Ottawa, Ont., Canada K1A055.
Contact: (For the standard) Director of Standards, CBEMA, 1828 L St. N.W., Washington, DC 20036; for CODASYL, Chairman, Programming Language Committee, Box 124, 928 Garden City Drive, Monroeville, PA 15146.
Application Area: Business data processing.

EL1
An extensible language which includes most of the concepts of ALGOL 60, LISP 1.5 but with an ALGOL-like syntax.
Application Area: Multipurpose.

FORTRAN (*FOR*mula *TRAN*slation)
The first language to be used widely for solving numerical problems. Still used primarily for numeric computations, but has been used in many other problem areas. Has existed in many versions and been implemented on almost all computers. The first language to be an ANSI standard. There are actually two standards: FORTRAN, which is essentially FORTRAN IV, and Basic FORTRAN, which is a proper subset of FORTRAN. An extensive revision of the existing 1966 ANSI standards is nearing completion, expected to be published late in 1976 or early 1977. The new standard includes the complete language and also a proper subset in a single document. The 1966 standard was also adopted as an ISO standard, and the revision is being considered by ISO to replace it. WATFIV is a widely used dialect of FORTRAN.
Implemented on: Almost all computers.
Ref: American National Standard, FORTRAN, ANS X3.9-1966; American National Standard Basic FORTRAN, ANS X3.10-1966; "Clarification of FORTRAN Standards—Second Report," *Comm. ACM*, Vol. 14, No. 10 (Oct. 1971); *FORWARD, FORTRAN Development Newsletter*, ACM SIGPLAN ad hoc Committee on FORTRAN Development c/o Loren P. Meissner, 50-B 3239, Lawrence Berkeley Laboratory, Berkeley, CA 94720.
Contact: (For the standard) Director of Standards, CBEMA, 1828 L St. N.W., Washington, DC 20036.
Application Area: Numerical scientific.

LEAP (*L*anguage for the *E*xpression of *A*ssociative *P*rocedures)
A language based on ALGOL 60 which contains set-theoretic and associative operations and data types.
Implemented on: DEC PDP-10, PDP-11 as part of SAIL.

Ref: Feldman, J.A., and Rovner, P.D., "An ALGOL-Based Associative Language," *Comm. ACM*, Vol 12, No. 8 (Aug. 1969); Rovner, P.D., and Feldman, J.A., "The LEAP Language and Data Structure," *Proc. IFIP Congress 1968*, Vol. 1, North-Holland Publishing Co., Amsterdam, Netherlands (1969).
Contact: Jerome A. Feldman, Dept. of Computer Science, Univ. Rochester, Rochester, NY 14627.
Application Area: Multipurpose.

LISP 1.5 (*LIS*t Processing)
A very sophisticated and theoretically oriented language for doing list processing. Various dialects exist, e.g., MACLISP, INTERLISP.
Implemented on: Many computers in both batch and interactive modes.
Ref: Berkeley, E.C., and Bobrow, D.G. (Eds.), *The Programming Language LISP: Its Operation and Applications*, M.I.T. Press, Cambridge, Mass. (1966); Weissman, C., *LISP 1.5 Primer*, Dickenson Publishing Co. Inc., Belmont, Calif. (1967).
Contact: ————
Application Area: List processing.

Logo
A recursive procedure-based language designed for educational applications. Used in teaching mathematics, heuristics, and formal methods in courses ranging from elementary to university levels.
Implemented on: Several DEC Computers and IBM System/360.
Ref: Feurzeig, W., et al., *Programming Languages as a Conceptual Framework for Teaching Mathematics*, BBN Report No. 2165 (June 1971); Papert, S., *Teaching Children Thinking*, AI Memo 247, M.I.T. Artificial Intelligence Lab., Cambridge, MA 02139 (Oct. 1971).
Contact: Wallace Fuerzeig, Bolt, Beranek & Newman, 50 Moulton St., Cambridge, MA 02138.
Application Area: Multipurpose.

MUMPS (*M*assachusetts General Hospital *U*tility *M*ulti-*P*rogramming *S*ystem)
A fairly general language with emphasis on string handling and complex data handling including hierarchical data. Developed and standardized by the MUMPS Development Committee which includes representatives from many hospitals, universities and goverment agencies. Active users group exists, and wide variety of applications have been developed using MUMPS, although primary purpose is for use in medical applications.
Implemented on: PDP-10, PDP-11.
Ref: MUMPS LANGUAGE STANDARD, Superintendent of Documents, U.S. Government Printing Office, NBS Handbook 118, SD Catalog No. C13. 11:118 (1975); MUMPS News (continuing issuance), 700 South Euclid Avenue, St. Louis, MO 63110.
Contacts: Joan Zimmerman, Executive Secretary, MUMPS Users' Group, 700 South Euclid Avenue, St. Louis, MO 63110 *or* Jack Bowie, Chairman, MUMPS Development Committee, Laboratory of Computer Science, Massachusetts General Hospital, Boston, MA 02114.
Application Area: Multipurpose.

PASCAL
A language designed to enable teaching of programming as a systematic discipline and to do systems programming. Based on ALGOL, emphasizing aspects of structured programming, and extending ALGOL primarily with convenient data structuring facilities. Has been used to write its own compiler.
Implemented on: Many computers.
Ref: Jensen, K., and Wirth, N., *PASCAL User Manual and Report*, Springer Study Edition (1975); Wirth, N., *Systematic Programming*, Prentice-Hall, Englewood Cliffs, N.J. (1972).
Contact: U. Ammann, Fed. Inst. of Technology, Clausiusstr. 55, 8006 Zurich, Switzerland; George H. Richmond, Univ. Colorado Computing Center, 3645 Marine St., Boulder, CO 80302.
Application Area: Multipurpose.

PILOT
A simple language for preparing computer-assisted instruction courses. Various versions have been written in BASIC, APL\360, and SNOBOL, and some have different names.
Implemented on: Several computers.
Ref: Rubin, S., "A Simple Instructional Language," *Computer Decisions*, Vol. 5, No. 11 (Nov. 1973); Rubin, S., *PILOT 73*, Stanford Research Institute, Menlo Park, CA 94025.
Contact: John Starkweather, Office of Information Systems, Univ. Calif., San Francisco, CA 94143.
Application Area: Computer-assisted instruction.

PL/I
A language suitable for doing numerical, scientific, and business data processing problems and for systems programming. Combines the most significant concepts from previous languages in the individual areas. ANSI standard nearly completed, expected mid-1976. Standard was developed jointly with ECMA, and is being submitted for approval by OSI also. PL/C is a widely used dialect of PL/I.
Implemented on: IBM System/360, Honeywell 6180, Burroughs, and CDC computers.
Ref: IBM System/360 PL/I Reference Manual, Form C28-8201, IBM Data Processing Division, White Plains, NY 10604; *Multics PL/I Language*, Document AG97, Honeywell Information Sustems, 60 Walnut St., Wellesley Hills, MA 02181; BSR X3.53 Programming Language PL/I (draft standard).
Contact: (For the standard) Director of Standards, CBEMA, 1828 L St. N.W., Washington, DC 20036.
Application Area: Multipurpose.

PPL (*P*olymorphic *P*rogramming *L*anguage)
An interactive, extensible language which contains facilities for defining new data types and operators.
Implemented on: DEC PDP-10 and PDP-11.
Ref: Taft, E.A., *PPL User's Manual*, Aiken Computation Lab., Harvard Univ., Cambridge, MA (Sept. 1974); Standish T.A.; *An Introduction of PPL Programming,* Aiken Comp. Lab., Harvard Univ., Cambridge, MA (Sept. 1974).
Contact: Thomas A. Standish, Dept. of Information and Computer Sciences, Univ. Calif. at Irvine, CA 92664 *or* Edward A. Taft, Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, CA 94304.
Application Area: Multipurpose.

PROTEUS
An extensible language with a small core from which to extend. Various descendants exist; see, e.g. PARSEC.
Implemented on: XDS 940; different descendants on IBM System/360, 370, UNIVAC 1108, SIGMA 5.
Ref: Bell, J.R., *The Design of a Minimal Expandable Computer Language*, Dept. of Computer Science, Stanford Univ., Stanford, Calif. (Ph.D. thesis), (Dec. 1968).
Contact: James R. Bell, Digital Equipment Corp., Research & Development, 146 Main Street, ML3-4/E41, Maynard, MA 01754.
Application Area: Systems programming.

SIMULA 67 (*SIMU*lation *LA*nguage, 1967)
A powerful language designed as a true extension of ALGOL 60, with built-in capabilities for list processing, text handling and simulation. Distinct from its predecessor known as SIMULA (now SIMULA I), which was primarily a simulation language.
Implemented on: Many computers.
Ref: Dahl, O.S., Myhrhaug, B., and Nygaard, K., *Common Base Language*, Norwegian Computing Center, Oslo 3, Norway; Birtwistle, G.M., Dahl, O.J., Myhrhaug, B., and Nygaard, K., *SIMULA begin*, Auberbach, Philadelphia (1973). Also, *Simula Newsletter* from Norwegian Computing Center.
Contact: SIMULA group, Norwegian Computing Center, Forskningsveien 1B, Oslo 3, Norway.
Application Area: Multipurpose.

SMALLTALK
A powerful extensible language designed for the Xerox Interim Dynabook. It is an object-oriented extensible language which draws upon and expands many of the ideas in Simula, Flex, Logo, and Lisp. While general purpose, it is strongly biased towards non-numeric computation.
Implemented on: Interim Dynabook.
Ref: Kay, A., and Goldberg, A., "Personal Dynamic Media," *Computer Magazine*, Vol. 10, No. 3 (March 1977) pp. 31-43; Kay, A., "Micro Electronics and the Personal Computer," *Scientific American* (Sept. 1977) pp. 231-244; Kay, A., and Goldberg, A., *Smalltalk-72 Instruction Manual*, Xerox PARC (1976).
Contact: Alan Kay or Adele Goldberg, Learning Research Group, Xerox PARC, Palo Alto, Calif.
Application Area: Interactive Programming.

SNOBOL4
A language emphasizing string handling and pattern matching.
Implemented on: Most large-scale computers.
Ref: Griswold, R.E., Poage, J.F., and Polonsky, I.P., *The SNOBOL4 Programming Language*, 2nd ed. Prentice-Hall, Englewood Cliffs, N.J. (1971); Griswold, R.E., and M.T., *A SNOBOL4 Primer*, Prentice-Hall, Englewood Cliffs, N.J. (1973); *SNOBOL Bulletin*.
Contact: Ralph E. Griswold, University Computer Center, Univ. Arizona, Tucson, AZ 85721.
Application Area: String processing.

SPEAKEASY
An easily learned yet powerful array processing language with built-in matrix algebra. Library-oriented system has facilities for statistical analysis, graphical output and database access. Is a powerful desk calculator in a time-sharing environment.
Implemented on: IBM System/360, 370, DEC PDP-10, and Japanese computers.
Ref: Cohen, S., *The SPEAKEASY-3 Reference Manual*, ANL-8000, National Technical Information Service, U.S. Dept. of Commerce, 5285 Port Royal Rd., Springfield, VA 22151; Cohen, S., "Speakeasy," *SIGPLAN Notices*, Vol. 9, No. 4 (April 1974).
Contact: Stanley Cohen, The Speakeasy Center, Argonne National Lab., Argonne, IL 60439.
Application Area: Numerical scientific.

TRAC® Language
Interactive string manipulation language involving nested functions and macro facilities. (®Trademark and Service Mark of Rockford Research, Inc.)
Implemented on: DEC PDP-10 and other computers.
Ref: Mooers, C.N., "TRAC: A Procedure-Describing Language for the Reactive Typewriter," *Comm. ACM*, Vol. 9, No. 3 (March 1966); *Definition and Standard for TRAC® T-64 Language*, Rockford Research, Inc., 140 1/2 Mt. Auburn St., Cambridge, MA 02138 (1972).
Contact: Calvin Mooers, address above.
Application Area: String processing. □

"Our Christmas card list was getting so large that we had to get one."

Z-80 PILOT

In previous issues (Volume 5, Numbers 5 & 6) we published an experimental version of a PILOT interpreter in Z-80 assembly language written by our longtime friend Dean Brown of Zilog. Although Dean has now made a number of additions to his PILOT, it still takes only 700 bytes in assembly code. The PILOT is designed to run on a system that has a Z-80 CPU, Zilog's disc operating system (which takes 16K of memory) and one or two disc drives.

With the exception of the compute command, C:, the Z-80 PILOT commands work as do those described on page 22. C: enables certain limited types of arithmetic so you can keep track of scores, number of guesses, and so on. There are four numeric variables: I, J, K, and L. Each variable may be set to zero or have the number 1 added to it.

C: ZI The letter Z before a numeric variable sets the value to 0; so I is now 0.

C: K A numeric variable *not* preceded by a Z causes 1 to be added to the value; so 1 is added to the value of K.

You may obtain a listing of Z-80 PILOT from Zilog Corporation. Or send a 7-inch soft-sectored disc to Zilog and it will be returned with a free copy of Z-80 PILOT.

Contact:
Dean Brown
Zilog,Inc.
10460 Bubb Road
Cupertino, CA 95014

PILOT

PROGRAMS BY ELLEN NOLD & SALLIE CANNOM

PILOT is a language that's easy to learn and easy to use. *People's Computers* regularly features PILOT articles and programs. Here are 3 computer-assisted instruction (CAI) programs in PYLON, a first cousin of PILOT; they were written by Ellen Nold and Sallie Cannom of Stanford University for use in freshman English composition classes.

On this page is what happened when a student named Chris 'ran' a program named 'Cinnamon'. Each line that Chris typed is preceded by '>'.

A summary of some PILOT commands:

- R: Remark indicated that the text following on the same line is a comment about the program, and is not part of the program.
- T: Type prints out on the terminal the text following on the same line.
- A: Ask causes the computer to pause and wait for user input.
- M: Match checks to see if what the user typed in for the last Ask command matches any of the items listed after the M: command. If there is a match, the match flag is turned on; if there is no match, the match flag is not turned on.
- J: Jump causes a branch to the label named: that command is executed next.
- Y: and N: Yes and No test the match flag. 'JY:' means 'jump if the match flag is yes, or turned on.'
- E: End stops execution of that part of the program in which it occurs.
- C: Compute permits evaluation of at least some arithmetic expressions.
- U: Use calls subroutines.

RUN OF CINNAMON

```
 Hi!  What's your full name?
>Christine Anne Brunet
 For what reason were you given a name?
>for purposes of identification
 Yes.  Your name identifies you.
 It signals your uniqueness.
 Your parents gave you their last name to show
 that, in a sense, you belong to them.  Your first
 and middle names reflect your specialness.

 When do people's last names change?
>women often change their names upon marriage
 Yeah.  Why is it that the woman's name
 changes in marriage?
>it's a cultural tradition
 Our tradition has it that a woman belongs to her husband.
 The name change indicates a shift in the power relationship.

 Speaking of tradition, why do you suppose God had Adam and Eve
 name all the animals instead of doing it himself?
>So they'd remember the name s better?
 Perhaps to give them a sense of being in charge.

 Names are powerful and naming is a powerful process.
 Members of certain African tribes, for example, always
 use an alias. Can you guess why?
>To permit them to tell their tru names only to friends
 They are fearful that harm may be done to them
 by someone holding the knowledge of their real name.

 Names also categorize us in a rather final way.
 Born Elmer you will probably remain Elmer.
 Some Native American tribes get around this "locking in" by giving
 tentative names to their offspring.  Later the children
 get other names.  What do you think of that?
>Prett interesting idea.
 I think it reflects one way of seeing people as process
 rather than fixed entities.

 So: names are identifiers; they are powerful;
 they sometimes signify ownership.
 Can you think of anything else?
>they can be a way of carrying on family traditions
 They can be descriptive and reflective of their object.

 As students of language, we need to remember
 that names and the naming process are arbitrary.
 The final authority to create and use names,
 to shape a reality by them, rests with you and with me.

 To explore your own name, try Sugar.
 Bye for now, Christine Anne Brunet.
```


MACE

```
R: Mace by Ellen Nold and Sallie Cannom  Stanford  9/73
T: Hi.  What's your name?
*Name A:
T: Mace is about facts or reports.  Perhaps you'd like to do
T: Marjoram before this program, but it isn't necessary.
T: Would you like to do Marjoram first?
A:
M: |NO|UH U|AIT GUESS NI
J N: @end@
T: All right, @name@, let's go on.
T: As you may remember, a fact or report is a statement
T: which can be verified or disproven by direct personal
T: observation, measurement or citations in reputable books.
T: What someone says about himself or herself is also a report.
T:
T: Is the statement, "He was an old, old man," a report?
A:
M: |NO|UH UI DON'T THINK|
T N: No.  That statement is not a report.
T Y: Right, @name@.
T: "Old," and other adjectives like "long," "wide," and "high"
T: do not describe the actual properties of the words they modify.
T: Rather, these words signal inferences about reports.
T:
T: Is "He was one hundred years old," a report?
A:
M: |NO|
T Y: Nope.  That's a report.
T N: Correct, @name@.
```

```
I:
I: Statistics and other "facts" must come from reliable sources.
T: Would you accept unquestioningly reports on campaign spending from:
T: Richard Nixon?
A:
Y: Sam Ervin?
A: The Justice Department?
A:
T: From your answers to these questions I could draw up a list
T: of the people and institutions named trusts.
T:
T: Of course, "reliability" is largely a personal judgement.
T: Reports can easily be slanted to fit a particular bias.
T:
T: For example, list the numbers corresponding to statements
T: which might appear in a newspaper article about a candidate
T: the writer didn't like:
T:  1. He's 6'7" - taller than the average American man.
T:  2. His shoulders are slightly stooped.
T:  3. His fingertips are stained.
T:  4. He smiled during the award presentation.
T:  5. His teeth are crooked.
T:  6. He wore a pressed suit.
A:
A:|2|
C Y: Set N1 to 1
N:|3|
C N: Set N2 to 1
N:|5|
C Y: Set N3 to 1
C T:=N1*N2*N3
J(T=3): right.
J(T=3): athrowd
J(T=0): 2 is correct.  Can you find others?
J(T=0): Set N1 to 1
I: (T>0): Good.  Are there others?
J(T>0): throwd
A:
M:|3|
C Y: Set N2 to 1
N:|5|
C N: Set N3 to 1
C T:=N2*N3
J (T=2): right.
J (T=2): athrowd
M: (T=0): 3 and 5 are the others.
J (T=2): athrowd
T: (T>0): There's one more.  Can you find it?
*Bow A:
T: 2,3, and 5 are the right answers.
*Throw T:
T: Accurate reports are hard to produce.
T: The bias (if any) of the "reporter" must be taken into account.
T:
T: If you'd like to work more on reports, and also on
T: inferences and judgements, try Marjoram.
T: Mustard is about details, and so is Sage.
T:
T: For a change of pace, try Sugar or Cinnamon.
T: Eye for now, dnamed.
T:
T:
*END END:
```

MACE continued

```
T: That statement is verifiable by public record.
T:
T: Is, "He says he's an old, old man," a report?
A:
T: That's a report.  It can be verified or disproven by witnesses
T: or by the speaker himself.
T:
T: O.K., dnamed, let's shift gears a bit.
T: Look around the room you're in.  Is there someone else there?
A:
M:|NO|UH U|
T Y: That means we'll have to jump to the next step.
J Y: #god
T: Good.  Type in a description of that person using only reports.
T: You've got two lines.
A:
A:
T: Hard to do isn't it?  Hope you didn't say something like
T: "She's a beautiful girl," or "He has long hair."
T: Those aren't reports.
T: "She has an upturned nose," and "His hair touches his shoulders" are.
T: Were all the statements in your description reports?
A:
M:|NO|UH U|I DON'T THINK|I'M NOT SU|
T N: Good.
J N: #god
T: Change those statements that might not be reports into reports.
T: You've got two lines.
A:
A:
T: All right, dnamed.  If you are confused about what a report is,
T: ask your instructor.
*Go T:
T:
T: One difficulty with reports (and with all kinds of statements)
T: is that you must trust the source.  Would you accept unquestioningly,
T: for instance, this statement from the victim of a mugging?
T: "It's sure he had hazel eyes and was about 5'6"."
A:
M: |NO|I DON'T THINK|UH U|
T N: Really, dnamed?
J N: #gold
I Y: Why not?
A:
*Go T: The trauma of the event might have impaired or distorted
T: the victim's perceptions.
```

Computer as *Art Critic*

BY JIM DAY

Which do you like better, Mona Lisa or Blue Boy? Why? Humanists are fond of asking such questions and arguing the relative merits of various works of art. There are some generally accepted rules of thumb regarding proper balance of form and color, but little seems to have been done to put the determination of artistic value on a scientific basis. As far as I know, no one has proven that such a determination is or is not possible. Most artists would probably say the idea of subjecting a work of art to scientific analysis is simply absurd. They may very well be right, but let's explore this question just a bit further.

Radio stations publish weekly lists of the top 10 records, ranked according to popularity. Isn't this a 'scientific' determination of artistic value? Of course! Well then, why not use a similar approach in determining the relative merit of artistic works in general? There is, no doubt, a subjective bias in such statistics since individual preferences are influenced by acquired tastes. Still, this approach may prove useful.

Another consideration is the fact that the complexity of a work of art or music may obscure the nature of the central theme. Do I like the William Tell overture because of the winds or the strings? Would I like it just as well or better without the second violin? Would I still prefer one painting over another if they were redone in black and white or altered in relative size?

I believe that the flexibility of the computer may be helpful in reducing the art of esthetic evaluation to a few simple principles (or at least proving once and for all that this cannot be done). One possible method would be to display 100 or so pairs of computer-generated patterns and ask a number of people to pick the one they like best of each pair. All possible permutations could be used to vary the pairing of patterns, so that each of the 200 patterns could be ranked by relative preference. By controlled variation of the graphic elements in each pattern one could learn whether people like three green blobs better than two red circles, for instance. The programming required would be quite simple, and the results might be interesting or even useful.

For example, this method could be used to find the optimum degree of randomness in a pattern. Perfect randomness has low artistic value because it doesn't look like anything except visual noise. Much of modern art falls into this category. On the other hand, perfect symmetry is high in basic artistic value but low in holding the interest of the viewer. How long can one look at a circle without getting bored? So there must be some optimum balance between randomness and regularity. In connection with this it's interesting that a circle, the ultimate in regularity, is the geometric expression of pi, a perfectly 'random' number.

Other things to be discovered are the best ratio of horizontal to vertical lines, long lines to short, straight lines to curves, and the most effective use of convergence, intersection, overlap, contrast, texture, color, brightness, etc. With a microcomputer driving a Cromemco Dazzler, for example, it should be fairly easy to experiment with these parameters to learn how each affects the statistical rating of a displayed pattern. If some general rules can be derived from such experiments, perhaps the notion of a computer as an art critic may not be so silly after all. □

There Ain't No User Science

A Tongue-in-Cheek Discussion of Interactive Systems

BY JACQUES VALLEE

Jacques Vallee takes computer professionals to task for their cruel and unusual treatment of Mr, Mrs and Ms average user. The author and his associates are at the Institute for the Future in Menlo Park, California. This article originally appeared in the 1976 Proceedings of the American Society for Information Science.

The current complexities of human interaction with computers can often be traced to poor design and to misunderstandings of the user's motivations and work patterns rather than to genuine scientific problems. Therefore, many problems related to poor user acceptance of computer systems do not suggest the need for a new branch of computer science, but for an examination of human factors from a psychological and sociological point of view. If there is a genuine 'user science,' it will have to be based on a much more sophisticated model of human thought processes than is currently available to systems designers.

This article draws from examples of man/system interaction in other fields to support the view that better standards, serious design guidelines, careful evaluation under real-world conditions and plain common sense would remove many of the existing obstacles for the computer user.

My role in this article is that of a black sheep. My objective is to raise some hidden issues and to question some of the obvious assumptions we are all making about user science. Computers can be expected to be used increasingly by non-specialists yet the interaction of this new user population with the systems we develop is not well understood, a need exists for a thorough analysis of man/machine systems. I am not questioning this need, but I am reluctant to see it glorified with the name 'user science,' at least until some of the major pitfalls have been recognized.

In the last three years, a small group of us has had the opportunity to observe interaction between several hundred non-specialists and various software systems used for 'conferencing.' Many of the participants in these conferences had never even used a computer terminal before and it was our responsibility to train them in everything from the setting of the switches on the device to the log-in procedure on the various networks they would be accessing, along with the various commands available to them through the program. In addition, some of them had to learn certain elementary operations of file management.

It has been an eye-opening experience for us to play this role, because the members of our group, who had varying degrees of previous acquaintance with computer systems, had lost their sense of perspective regarding the constraints placed by systems designers on the dialogue between the machine and its users. We knew that this interaction was a much-neglected area of systems programming, but we had never realized just how cumbersome, preposterous or plainly stupid the interfaces were.

Observing the user's frustration with such programs, outsiders might well conclude that an extremely complex area of research has been uncovered and that it represents a new science. They might even approve of research funds being spent to understand why nobody uses the information systems that research funds have been spent to implement. I will argue that such a conclusion is wrong, that it only lends dignity to a state of confusion that was unnecessarily *created by programmers* and should be *cleaned up by programmers.* If anything needs to be researched, it is not the user but the social structure that surrounds any information system, be it a set of rules and laws, a library or a computer-based facility. It is this social structure which dictates the interface, and to focus attention solely on the problems of the user is to miss the real issue.

WHY DON'T PEOPLE USE COMPUTERS?

Let me offer a simple starting point, suggested to me by my collegue Hubert Lipinski: 'Why don't people use computers?' *We* use computers, of course, and many of our friends do. But we represent a very small minority. We depend on computers for our work and for the information that guides many decisions we make. But people outside this community do *not* use computers in spite of the valuable services they can provide. Not only is the man-in-the-street disgusted, intimidated, awed or repelled by computers, but many professionals view information processing with hostility. The reasons are not buried deep in the dark recesses of the human mind. They are plain and simple. They begin with the terminal, which has a plethora of design problems. They grow with the difficulty of logging into a network or into a central computer. They blossom as soon as humans begin interacting with a piece of software.

The Terminal. Figure 1 shows a standard terminal keyboard. This is the kind of equipment our project has been shipping to schools, research institutions and government facilities around the country to train *people* (not programmers) to use our software. Let us assume that the terminal is provided with paper, is in perfect working order and that all the settings are correct. (I know this may sound trivial to you, but have you tried to explain to someone *over the phone* how to load a roll of heat-sensitive paper through a slit on a portable terminal? Do you know why there is an INTERNAL switch on a device that has a built-in coupler?)

Any *ordinary* person looking at this keyboard will assume that hitting SHIFT-E will result in the device typing the combination 'ENQ', because it appears on top of the E key. Thus, many of our users go through life without ever typing a capital letter for fear of unleashing some strange combination of letters or of starting some uncontrollable chain reaction. Of course, *extraordinary* persons like us know better. We know that SHIFT-E on this machine will type a normal capital E and that to obtain ENQ one has to hit CTRL-E. *It is knowledge like this that makes us extraordinary.* (I confess I have no idea of what would be the circumstances under which I might need to utter ENQ, and I would be honored to meet someone who does.)

The presence of these strange-sounding codes on the keyboard is a small problem. Worse is the fact that touch typing is inhibited by the shape and position of the keys, the variable-delay echoing, and the 'roll-over' feature that prevents depressing more than one key at a time. These features effectively decrease the abilities of the proficient typists.

Other problems associated with various terminals are: the glare on the keyboard, the glare of the plastic plate which covers the paper, the printing head which often obstructs the user's view of the last few characters, the placement of the keys and the intimidating row of function keys such as INT, BRK, or READY which have names that bear a very distant relationship to what happens when you push them.

The Log-In Game. We now peep into the living room where Mr and Mrs Average User, having finished dinner, decide to join their favorite computer network, as they have done every day for the last two years. Although the network has thus recorded nearly 800 interactions with them, it apparently has no way to remember their real names, so they have to be known by the code 'CROWN'. (This is a thrill to Mrs Average User, who is a spy movie fan, but Mr User can never remember the password.) Figure 2 shows what happens between the time when they hear the BEEP on the telephone and the time when the terminal types out something directly related to their interest, namely the message 'WELCOME.' We have repeated this operation under four separate commercial networks, which apparently stay in business in spite of their log-in procedures. (It would be unfair to this audience to include military or academic computer systems as examples. Their users have no choice.)

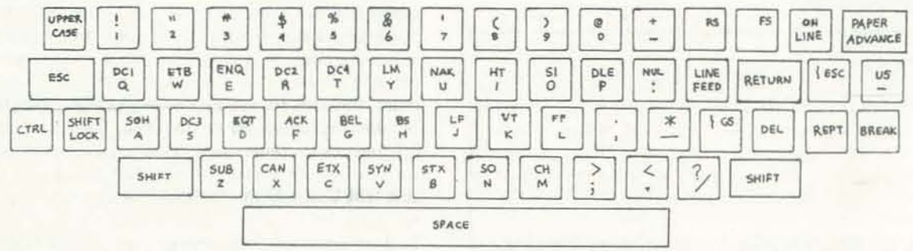


FIGURE 1. Poor Keyboard Design

```
TELENET: CR
         CR
  † 1    TELENET
  † 2    415 DK1
    3
    4    TERMINAL=T125 CR
    5
    6    @c 617 20c CR
    7
    8
  † 9    617 20C5 CONNECTED
   10
   11
  †12    BBN-TENEX 1.34.5, BBN-SYSTEM-C EXEC 1.54.12
   13    @LOG CR
   14    (USER) CROWN CR
   15    (PASSWORD) _____ CR
   16    (ACCOUNT #) CR
  †17    JOB 23 on TTY 151 19-JUL-76 13:15
  †18    PREVIOUS LOGIN: 19-JUL-76 11:03
   19    @RUN PROGRAM CR
   20
   21    WELCOME.
```

Figure 2. Log-In Procedure on A Commercial Network. Each † indicates an entire line that is meaningless to most users. The underlined portions were typed by the user.

For each interaction case, we have recorded three simple quantities:

- α: the number of lines appearing on the terminal before Mr and Mrs Average User see something meaningful;
- β: the number of keystrokes they need to type and
- δ: the number of characters typed by the system having no meaning or relevance to their interest.

The reader will see in Table 1 that even on the most expeditious network, INFONET, it takes 29 keystrokes to do nothing. (How would you like to dial 29 digits every time you want to call the girl next door? Telephone companies go out of business over things like that. Computer users go out of their minds.) On TELENET, it takes 48 strokes to do nothing; but one is rewarded with a deluge of messages in reply, ranging from '415 DK1' to 'JOB 23 on TTY151' with rather obscure meanings, especially if you know what a TTY looks like. (Note that following TERMINAL=, the user replies T125. This is a code that stands for TI725!)

	TELENET	TYMNET	CYBERNET	INFONET
α:	20	10	14	8
β:	48	35	46	29
δ:	116	10	95	54

TABLE 1. Obfuscation Parameters in Four Commercial Networks

The Dialogue. Now that Mr and Mrs Average User have won the log-in game, they can run their favorite program. They have graduated to the wonderful world of software where anything can happen. They could, for instance, get the message 'DRUM FULL,' after which the terminal will refuse to do anything. Mr Average User, who didn't even know he *had* a drum, is extremely impressed. (I recall being introduced to a medical researcher in Europe a few months after a teleconference through which we had first 'met'. He was a calm Britisher whose first words to me, delivered in a kind but a reproachful tone, were, 'You know your message that says 'HOST DIED'? Well, wouldn't it be more kind to say 'HOST PASSED AWAY'?')

Indeed it is in death that computers come closest to imitating humans. The phenomena that accompany their last few seconds are as sad, emotional and messy as those of their human masters. In their bereavement, Mr and Mrs Average User might well despair of ever finding their computer healthy again, as was the case with my friend, Bob Johansen, when he lost a program amidst much typing of strange characters (Figure 3). Since he had come to the Institute from a background in the sociology of religion, he assumed that the system was speaking in tongues. Yet, when he tried to reestablish contact, he was told that 'host' was no longer responding.

```
                    (TO FORUM)
                    OLY 7S?/ : ZI I CAMY$↑HV80/S81&9SJ38
                    _
Computer            HELLO 322 *: 2
dies here           @L 69

                    LATEST NEWS: 5 NOV
                    HOST NOT RESPONDING
```

FIGURE 3. Famous Last Words of a Dying Computer

The folklore surrounding the little peculiarities and idiosyncracies of each system is an amazingly diverse variety of colorful tales. Every network has its own way of rejecting you, of pulling you into strange traps, of making you wait or repeat what you have just said, of dying. Although such peculiarities can be fun for the boys and girls in the machine room, they have a devastating effect on people who are trying to get a job done. When the job involves communication and joint effort among groups located all over the Western hemisphere, as our teleconferences sometimes do, the results can be disastrous. Motivating the users again after every failure is a difficult and frustrating task. When we performed an analysis of questionnaires returned by participants in some early conferences, we found that two factors far outweighed all others in accounting for their reactions to our system. These two factors were *difficulties with terminals* and *fear of a network failure.*

THE OBFUSCATION IMPERATIVE

If there were a user science, the Obfuscation Imperative would constitute its first law. There is a short but interesting diagnostic on TELENET. It reads: 'SUBPROCESS UNAVAILABLE.' The puzzled user refers to the section called 'Explanation of Network Messages' of the user manual and reads:

> The specific host process included as part of the address is not available.

Note that the so-called 'explanation' has now unnecessarily added two new sources of confusion, the term 'host' and the word 'address.' The user has no recourse but to make an appointment with a system expert. The expert knows the true meaning of the message: the computer is down! Of such knowledge expertise is made. This is not user science. This is *obfuscatology*, the science of hiding things away from other people. It reaches a peak with system completion code 213 on the IBM 360. If your program dies with 'COMPLETION CODE 213–04,' you must consult the appropriate manual under IEC 132 I. The system expert who owns the manual will then read the expanded text of the explanation to you:

> The format 1DSCB for the data-set could not be found on the first volume (or the volume indexed by the volume sequence number) specified by the DD statement, or an I/O error occurred reading the F-1 DSCB for the data-set.

What the message means is simply: 'DATA-SET WAS NOT THERE.' Why didn't they say this in the first place? The answer has nothing to do with the computer: this message has exactly the same length as 'COMPLETION CODE 213-04,'' and the machine couldn't care less what the message says.

Is such evidence of deliberate obfuscation limited to our field? On the contrary, *it can be found whenever a community of specialists attempts to protect a privilege.* For example, in the 13th century, a surgeon named Arnold of Villanova recommended to his collegues to preserve their linguistic distance under the greatest diagnostic stress:

> Say that the patient has an obstruction of the liver, and particularly use the word '*obstruction*' because they do not understand what it means, and it helps greatly that a term is not understood by the people.

Examining current medical literature, Michael Chrichton concludes that contemporary physicians are still following the rule: they are trying to 'astound and mystify the reader with a dazzling display of knowledge and scientific acumen.' He also observes that most doctors ignore papers outside their own specialities because they can't understand them. Does medicine need a user science?

The legal profession, too, follows the obfuscation rule. When I bought my house, I signed a paper which reads:

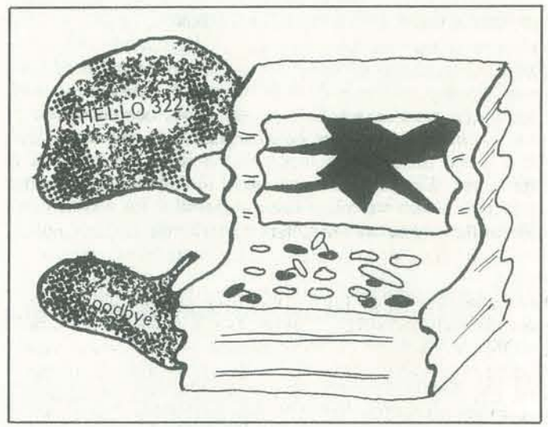
> This Deed of Trust applies to, inures to the benefit of, and binds all parties hereto, their heirs, legatees, devisees, administrators, executors, successors, and assigns.

Even photographers obfuscate. Witness this extract from the user manual for my modest camera:

> Using the flashmatic system, after you set the guide number,the correct exposure is automatically calculated as you focus. No more fumbling through lengthy calculations to find the proper F/stop: set the mark on the F/stop ring to the center index. Obtain the proper guide number by multiplying F/stop number times distance. For example, if ASA 80 film is being used, set the film speed on the calculator to ASA 80 and check the proper F/stop number at a distance of 10 feet.

In a common practice of software design, the same operation is given different names, depending on the particular subsystem! Here again, better training or a more detailed model of the user will not help. What is needed is a thorough clean-up of the command language. Under the KRONOS operating system of Control Data Corporation, for example, there is a command called 'LIST' which produces a listing of a file. Having obtained a listing, the user might want to edit the file. Under the EDITOR command, however, the LIST command becomes invalid. Only an expert will be able to tell you that the proper command to use now to produce the *same* listing of the *same* file is PRINT. But once you get out of the EDITOR command, naturally, PRINT is no longer recognized!

We could devote a whole session or a whole conference to such examples.



THE WIDE ANGLE FALLACY

If there was a user science, its second law could be called the Wide Angle Fallacy. When a disgusted user goes back to the designer with the statement, 'Your system doesn't perform the special function I need,' the designer's ego is deeply affected. To regain the good graces of his customer -- and to reestablish his or her own self-esteem -- the designer is likely to answer, 'I can fix it. I will add another command for you.'

Later, the same designer will be seen at conventions, meetings, and workshops, extolling the virtues of his system, the 'power' of which can be measured by the great number of commands it can execute. I believe this is often a fallacy and that both designers and users should recognize it. There is a similar fallacy in astronomy, related to the size of a telescope. Most

novices and many astronomy students believe that you see more in a big telescope than in a little one. The opposite is true, of course. Increasing the diameter of the telescope may collect more light, but it narrows the field of view. The analogy with software design is appropriate. The early versions of our FORUM conferencing system had dozens of commands, hence a lot of potential 'power', but few users could remember the whole structure. Careful monitoring of usage pointed to ways of simplifying it, and we started looking for opportunities to cut down the list of commands. For the last two years, we have been offering a teleconferencing service which gives the user a repertory of only six commands. The potential user population (which is analogous to the 'field of view' of a telescope) is now much larger. At the same time, we have found that no serious loss in 'power' had been experienced. On the contrary, the new, simpler commands correspond better to the basic primitives of the interaction we are trying to support.

What is true of systems programmers applies equally well to librarians who think they are expanding the 'power' of an index when they add more keywords and more terms into it. The whole issue of how to support the process of discovery instead of mimicking its side effects lies solidly buried under dozens of documentation systems which our profession is accumulating as a buffer between the scientist and his data.

IF NOT A USER SCIENCE, WHAT THEN?

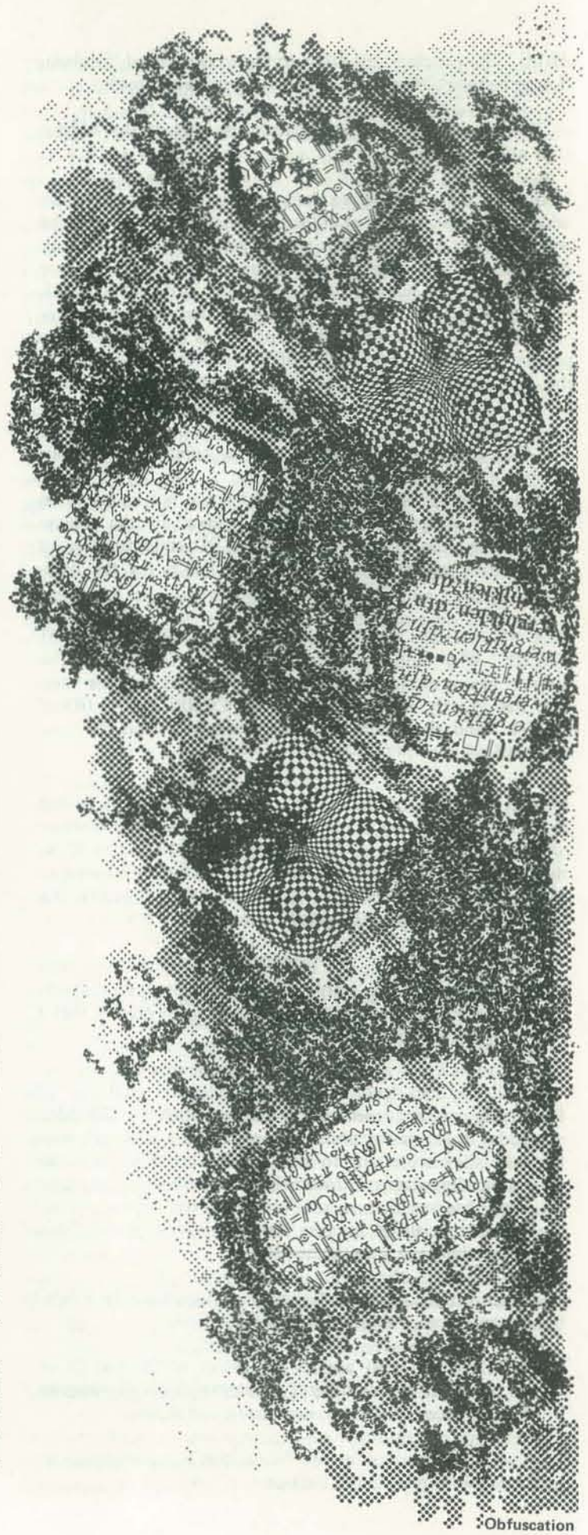
These observations do not make me feel an urgent need for a new science that would study the user's relationship with computer systems where even the most obvious steps to human interactions have been neglected. I will argue later that a real user science exists, but it is not to be found at this level. What we need instead is to follow a few guiding principles which will place those responsible for a system in a position to anticipate many user frustrations:

1) The first principle would be never to start implementing a system until the end users have been identified and given easy access to the designers.

2) The second principle would be to monitor everything, noting, however, that not all information is quantitative and that one cannot evaluate the potential impact of missing features.

3) A third principle would be to release systems to 'real users' as soon as their utility is apparent to them. A real user is characterized by the consequences for the user and for the designer if the service is not performed, by the fact that the service is funded with money which could be used in other ways and which comes from an operational (versus research or exploratory) budget and by an ultimate evaluation of the service in terms of an external outcome.

4) The fourth principle would be never to demand the user to type an input that is not relevant to the task at hand and never to give him an output that is outside the task context. (Our software now tries to intercept all system panic messages, replacing them with the statement: 'Sorry, we are having trouble with the computer.')



Obfuscation

Attempts to make computer professionals aware of the need for these elementary guidelines often end up in frustration. They assume that if you raise such a question, you can only do so out of either resentment or ignorance. At best, they will send you on your way with an elementary tutorial on operating systems and a PL/1 manual. This attitude was illustrated to us by a recent discussion between Thad Wilson, a member of the Institute staff who has used computers for 15 years, and the manager of the Northern California facility for a leading network company.

Wilson: 'Many of our users do not understand why the system sometimes suddenly stops and says, 'PLEASE LOG-IN.' How are they supposed to understand that? *They've already logged in.*'

Manager (trying to confuse Wilson): 'They need to log-in again because a failure has occured. It could be the node, the supervisor, the host, or the network.'

Wilson (not confused)' 'Why don't you simply tell them that you're sorry that service will be interrupted for a while?'

Manager: 'You don't realize how long it takes us to change something. We've been in business a long time. You should train your users better and give them our manuals.'

Wilson: 'I don't think it's a good idea. You should change the system instead.'

That actual discussion illustrates a sad point: when we encounter problems with the use of computer systems, we can change either the system or the people. What is frightening is that the computer industry probably has enough power now to start changing people. This network manager has already been changed, and he is working hard to change others. The programmers under him work hard to become like him. To modify the system is too difficult for them to even consider. The reasons they give are not technical, but social and bureaucratic. After all, they 'have been in business a long time.' Somebody would have to rewrite all those forms and all those manuals.

The social environment and the anecdotes surrounding the marketing, maintenance, and breakdowns of computer products should also come under the attention of would-be user scientists. The vast distance between the services delivered by the computer community and the users it pretends to serve was made obvious to us in the course of a computer conference which included as a participant Mr Richard Bach, the author of *Jonathan Livingston Seagull*, who lives in Florida. After a period of unusual silence from him, we learned that his terminal had broken down. We tried unsuccessfully to put pressure on network and terminal suppliers to try to help him. When contact was reestablished (a week later), he gave us some of the details:

> Private message from BACH 1-Jul-75 7:08 PM
> Some notes in the quiet here about events during the breakdown last week of my terminal. The interesting learning for me is that an individual with his terminal inoperative is the low priority item in the system. Good reasons, and all that, but the feeling in the computer termnal world is that good excuses are acceptable, which would be rare in a high-competition field.
>
> Private message from BACH 1-Jul-75 7:18 PM
> This system with the enormous potential for communication, education, entertainment, and intellectual atom-swapping is pressurized in a gas more inert than pure nitrogen. So far, I haven't even found somebody who will sell me a roll of paper for this thing (though I may have a line on one now). Headquarters people say, 'Call the local office,' not knowing their offices here are all closed or empty answering services. They promised to send me two rolls a week ago, which haven't yet arrived . . . I borrowed two from the repairman, who needs them replaced when mine ever arrive. . . .Somehow he is not allowed to sell them.

It would be most interesting to analyze user profiles, command usage and interaction patterns when computers begin to be used by writers and communicators of the caliber of Mr. Bach. But first we must find a way to provide them with paper for their terminals and give them reliable access to systems they can use. This simple prerequisite has not yet been met by the computer community.

The examples quoted above have attempted to show that situations frustrating to the nonspecialist user have often been deliberately created to protect a self-styled elite of programmers. A massive effort to redefine the job control language (JCL) of IBM, for instance, would free the creativity of thousands of people and expand the market for many computer services; but at the same time, it would make obsolete a mass of folklore and machine room recipes that passes for knowledge and creates job security for programmers throughout the world. The Obfuscation Imperative is a clear marketing strategy. Someone trying to understand the relationship of the user to the system has to first realize that the situation is dictated by social, psychological, economic and professional constraints. Only when elementary design – such as those we have only begun to outline here – have been applied to software interfaces will we be able to really speak of a genuine science.

The topics which could be studied as part of a design science are varied and exciting. They are already found in the investigation of cognitive styles (as in the work of Peter Keen at Stanford), the perception of the computer as a confidant or an adversary (as in Chris Evans' experiments in England), several artificial intelligence prototypes, mapping, interactive graphics, computer-aided instruction and advanced data analysis packages. This experience, however, has never been pulled together into a body of knowledge; it is certainly desirable and urgent to do so, but it still leaves unanswered the question of a general user model.

In the meantime, I can only concur with William Blake who wrote in *Jerusalem* (f10.20):

> *I must create a system*
> *Or be enslaved by another man's!* □

FORTRAN MAN

BY LEE SCHNEIDER & TODD VOROS

When last we left Our Hero, it would appear that he had gotten himself into a somewhat less-than-heroic position . . . When informed that Doktor Debug's beautiful daughter Parity had fallen into the clutches of the evil Count Algol, Fortran Man immediately went compute-bound to rescue her . . . By tracing the flow of the Count's decremented assistant, Igor the File Snatcher, F-Man discovered the relocated location of the Count's hideout within the great complex of the now deserted Von Neumann beer works . . . and there confronted the evil Count! The Count fled the battle . . . but not before triggering a hidden trap address . . . and only the fastest of action on the part of Our Hero allowed Parity to be preserved from a fate worse than intermittant glitches! With Parity maintained and Igor fatally compressed in the fight, F-Man pursued the Count to the dark, mysterious Castle Algol, and rushes in, ready to do battle . . . but then F-Man is Trapped! Caught by the falling edges of a concealed gate output, his outputs are held in a low state, as streams of deadly Disassembly Gas source from a multitude of small, concealed ports at ground level!

And try as he might, there seems to be no EXIT point for Our Hero . . . and the evil Count advances slowly on his helpless victim . . . though careful to remain out of the drifting modules of deadly Disassembly Gas!

An amazed Count signals the ports to close, then increments foreward quickly to examine the now empty trap . . . but then, a familiar vector CALLs out from behind him . . .

But then, instead of crumbling into a compile of disassembled bits, the trapped figure suddenly begins fading away . . . and vanishes with nary a trace routine!

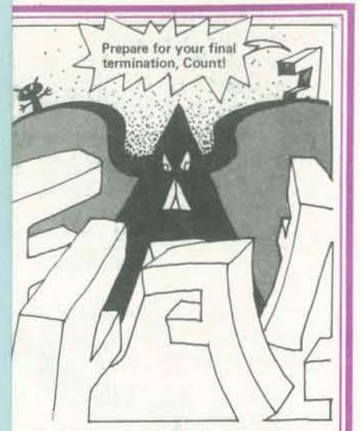
. . . and the Count steps back in sudden surprise, for what he sees behind him is . . .

The Count quickly reverses direction . . . and there behind him on an upper level, stands an all-too-familiar figure!

With nary a wasted motion and with a glint of evil in his I/Os, the Count increments rapidly toward his foe . . . and then suddenly reverses as another CALL vectors in from the other side of the room . . .

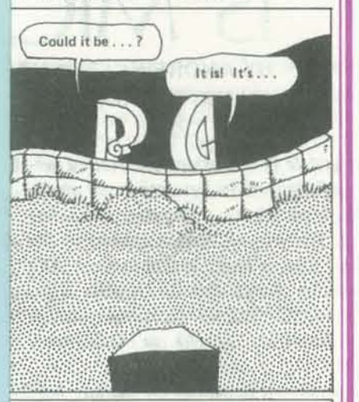
The Count is immobilized . . . stunned into a WAIT state! For what he sees before him is not just ONE Fortran Man . . . but TWO!

Holiday Subscriptions & gifts



And with that . . . the great battle begins!

With the interrupt flag in place, the exit point of the castle is suddenly opened . . . and a single, long figure branches out . . . heading swiftly across the data field towards them . . .



And a microsecond later, Castle Algol explodes into a billion bytes!

Is this the final END for the evil Count Algol?

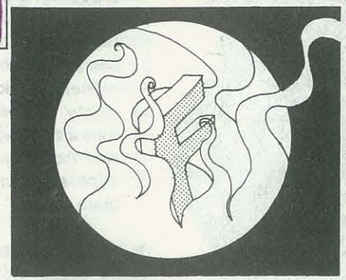
Has the Count finally been terminated?

Join us for the final episode next issue . . . and find out!

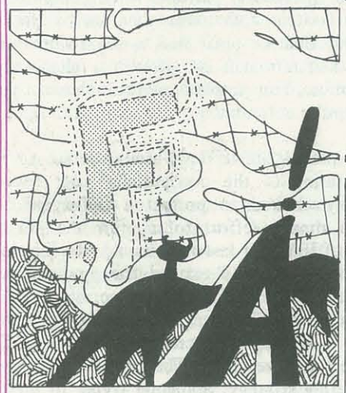
FORTRAN MAN

BY LEE SCHNEIDER & TODD VOROS

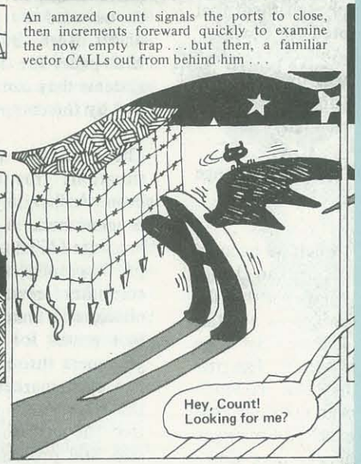
When last we left Our Hero, it would appear that he had gotten himself into a somewhat less-than-heroic position . . . When informed that Doktor Debug's beautiful daughter Parity had fallen into the clutches of the evil Count Algol, Fortran Man immediately went compute-bound to rescue her . . . By tracing the flow of the Count's decremented assistant, Igor the File Snatcher, F-Man discovered the relocated location of the Count's hideout within the great complex of the now deserted Von Neumann beer works . . . and there confronted the evil Count! The Count fled the battle . . . but not before triggering a hidden trap address . . . and only the fastest of action on the part of Our Hero allowed Parity to be preserved from a fate worse than intermittant glitches! With Parity maintained and Igor fatally compressed in the fight, F-Man pursued the Count to the dark, mysterious Castle Algol, and rushes in, ready to do battle . . . but then F-Man is Trapped! Caught by the falling edges of a concealed gate output, his outputs are held in a low state, as streams of deadly Disassembly Gas source from a multitude of small, concealed ports at ground level!



An amazed Count signals the ports to close, then increments foreward quickly to examine the now empty trap . . . but then, a familiar vector CALLs out from behind him . . .



But then, instead of crumbling into a compile of disassembled bits, the trapped figure suddenly begins fading away . . . and vanishes with nary a trace routine!

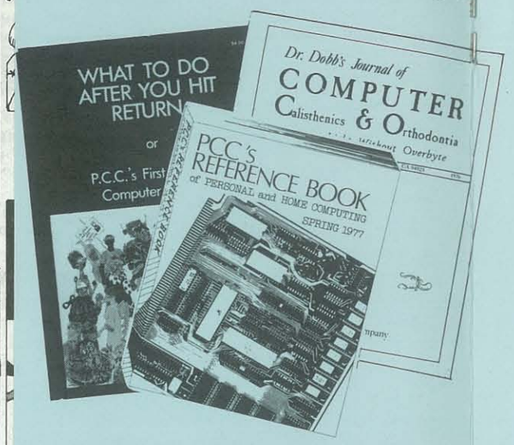


With nary a wasted motion and with a glint of evil in his I/Os, the Count increments rapidly toward his foe . . . and then suddenly reverses as another CALL vectors in from the other side of the room . . .

. . . and the Count steps back in sudden surprise, for what he sees behind him is . . .

SEND A DIFFERENT GIFT FOR THE HOLIDAYS!

Give a gift subscription (or two!) to *People's Computers* this holiday season. For only $8 you can give a gift that keeps coming all year round. And every issue of *People's Computers* will bring exciting and up to the minute articles, listings, interviews, games, announcements, and more. Use the card opposite to order subscriptions for yourself and friends. Send in your gift subscriptions immediately and we'll mail special gift announcement cards in time for the holidays.



Dr. Dobb's Journal – Volume One
Now you can purchase a single bound volume containing all ten issues of the first year of **Dr. Dobb's Journal**, described as "THE software source for microcomputers. Highly recommended," in *The Data Bus.* **Dr. Dobb's Journal** contains no paid advertising, and its primary purpose is "to place significant software into the public domain". $13.

WHAT TO DO AFTER YOU HIT RETURN or P.C.C.'s First Book of Computer Games
A fun gift book of 48 computer games written in BASIC – strategies, treks to the stars, simulations, graphics, wumpus hunts, and much more. This educational book is "crammed to the margins with interesting tidbits". $8.

P.C.C.'s Reference Book of Personal and Home Computing
An invaluable gift that anyone interested in home computing would appreciate, containing both survey and nuts and bolts articles for the experienced and the no-so-experienced user, the complete documented source and object code for a 2K Tiny BASIC, reference material: from bibliographies to lists of stores, companies, and hobbyist clubs, and a massive index of articles in ten major hobbyist magazines. 248 pages, $5.95.

people's computers

New & Gift Subscriptions

Send a gift subscription to *People's Computers* this holiday and for only $8 give a gift that keeps coming all year round. Mail this form today and we'll send a gift announcement card in time for the holidays.

SEND A GIFT SUBSCRIPTION TO:

1. NAME _____
   ADDRESS _____
   CITY/STATE _____ ZIP _____
   32
2. NAME _____
   ADDRESS _____
   CITY/STATE _____ ZIP _____
   32

Sign my gift card: _____

AND BILL ME:

NAME _____
ADDRESS _____
CITY/STATE _____ ZIP _____
35

☐ START A SUBSCRIPTION FOR ME
☐ Visa/BankAmericard Card Number _____
☐ Master Charge Expiration Date _____

(Foreign rates available on page 2.)

GIFTS

PLEASE SEND ME THESE HOLIDAY GIFT BOOKS:

_____ copies of **PCC's Reference Book** for $5.95 plus 95¢ for shipping and handling (California residents please add 35¢ sales tax) per copy.

_____ copies of **WHAT TO DO AFTER YOU HIT RETURN** for $8 (California residents please add 48¢ sales tax) plus $1 for handling per copy.

_____ copies of **Dr. Dobb's Journal Volume One** for only $13 (California residents please add 78¢ sales tax) per copy.

Mail this card immediately so you'll get the books in time for the holidays.

BILL AND MAIL TO:

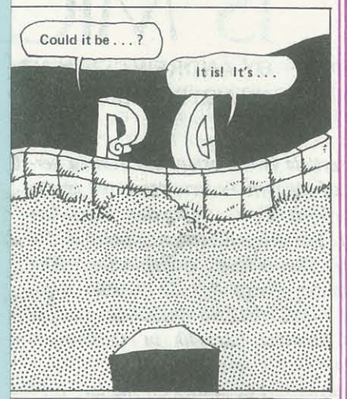
NAME _____
ADDRESS _____
CITY/STATE _____ ZIP _____

☐ Visa/BankAmericard Card No. _____
☐ Master Charge Expiration date _____

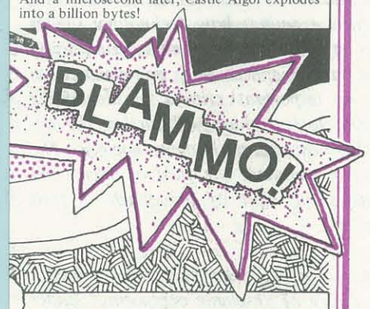
(All overseas orders will be sent airmail unless otherwise specified.) 3X

And with that . . . the great battle begins!

With the interrupt flag in place, the exit point of the castle is suddenly opened . . . and a single, long figure branches out . . . heading swiftly across the data field towards them . . .



And a microsecond later, Castle Algol explodes into a billion bytes!



Is this the final END for the evil Count Algol?

Has the Count finally been terminated?

Join us for the final episode next issue . . . and find out!

FORTRAN MAN

BY LEE SCHNEIDER & TODD VOROS

When last we left Our Hero, it would appear th
he had gotten himself into a somewhat le
than-heroic position . . . When informed th
Doktor Debug's beautiful daughter Parity h
fallen into the clutches of the evil Count Alg
Fortran Man immediately went compute-bou
to rescue her . . . By tracing the flow of t
Count's decremented assistant, Igor the F
Snatcher, F-Man discovered the relocated lo
tion of the Count's hideout within the gre
complex of the now deserted Von Neuma
beer works . . . and there confronted the e
Count! The Count fled the battle . . . but n

But then, instead of crumbling into a compi
of disassembled bits, the trapped figure sudden
begins fading away . . . and vanishes with nar
a trace routine!

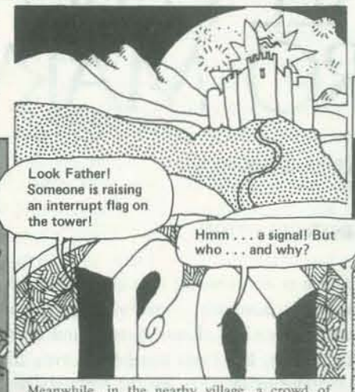
With nary a wasted motion and with a glint
evil in his I/Os, the Count increments rapid
toward his foe . . . and then suddenly revers
as another CALL vectors in from the other si
of the room . . .

And that's not all! For in a microsecond, Castle Algol is suddenly overflowed with Fortran Men!

The battle rages on . . . with the tremendous strength of the Count pitted against a multitude of compiler copies! The entire structure of Castle Algol resonates with the oscillations of the battle . . . and the Count is much too compute-bound to notice the single F-like figure branching away from the battle and disappearing up the voltage ramp to the upper levels . . .

The figure reaches them quickly . . . and, performing an unconditional JUMP over the piles of data blocks, joins the Doktor and Parity . . .

And as the evil Algol tries to take account of the situation, Fortran Man explains in a methodical voice . . .

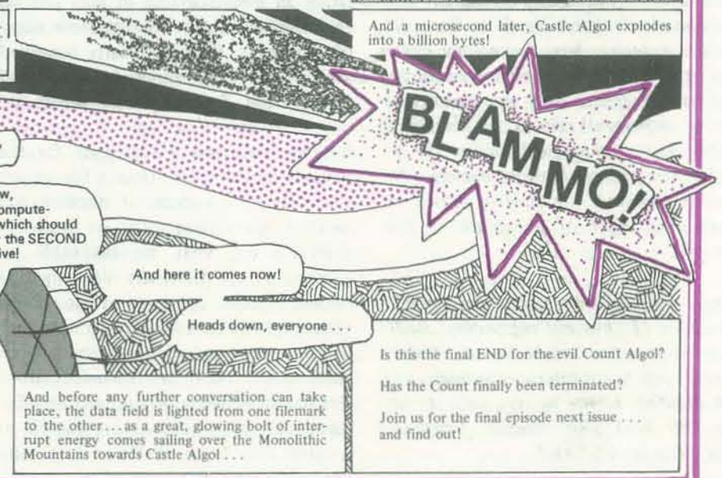


Meanwhile, in the nearby village, a crowd of curious files have gathered at the edge of the data field . . . attracted by the sudden oscillations at Castle Algol. The good Doktor and Parity are among them, and as they watch . . .

And with that . . . the great battle begins!

With the interrupt flag in place, the exit point of the castle is suddenly opened . . . and a single, long figure branches out . . . heading swiftly across the data field towards them . . .

And a microsecond later, Castle Algol explodes into a billion bytes!



And before any further conversation can take place, the data field is lighted from one filemark to the other . . . as a great, glowing bolt of interrupt energy comes sailing over the Monolithic Mountains towards Castle Algol . . .

Is this the final END for the evil Count Algol?

Has the Count finally been terminated?

Join us for the final episode next issue . . . and find out!

People will soon be meeting micro-computing devices everywhere.

IF 'SMALL IS BEAUTIFUL' IS MICRO MARVELOUS?

BY ANDREW CLEMENT
ILLUSTRATIONS BY RENNIE WISWELL

This article is a substantially revised version of a paper Andrew presented at the First West Coast Computer Faire last April. The original version appears in the Faire's Proceedings, *available from The Computer Faire, Box 1579, Palo Alto, CA. 94302 ($12.68 including postage and handling; $13.40 in California).*

The article's title draws on the late E.F. Schumacher's book Small is Beautiful, *which examines how technology shapes society. He identifies smallness, simplicity, capital cheapness and non-violence as being important criteria in identifying technologies which will further the development of an humane society. The article examines the field of microcomputing in light of Schumacher's four criteria.*

The paper was written in connection with a survey of "humane computing" under a grant from the Canada Council. Andrew invites you to contribute experiences and/or thoughts to the survey: write to him at 789 West 18th Avenue, Vancouver B.C., Canada V5Z 1W1.

INTRODUCTION

There is no question that advanced technologies have had a profound effect on modern society. Every person and every institution has been touched in even the most intimate ways by the sophisticated tools that abound in our society. We have been spectacularly successful in translating an understanding of Man and his environment into tools that serve many of our needs. We have clearly benefited greatly and now enjoy freedoms and powers that few have had before us.

But we have paid a high price for the rewards of technical mastery for we are also reaping a harvest of negative and harmful side-effects. We are seriously depleting our vital, non-renewable resources; we are drastically disrupting the natural systems that support life on this planet; and we have broken so many traditional social bonds that now many of us stand alone, afraid, and alienated. All of this has happened largely unintentionally, merely as part of the sincere pursuit of a better life. These ill-effects are usually unforeseen and often come well after the enjoyment of the initial benefits.

Hindsight is of course close to 20-20 while the future is usually pretty murky. However, to the extent that we can anticipate social impact of a technology as potent and complex as micro-computing, it would seem advisable to explore its implications and try to chart a course that gives us that maximum benefit over the long run.

Micro-computers have not yet begun to affect many people's lives directly but there is good reason to believe that in the foreseeable future their impact will be great. We are witnessing at the present time the rapid emergence of the computer from within the confines of large institutional settings and into the view of the general public. Until now most people have not really seen a computer or had to deal with one directly. Rather they have learned of them by report and communication with them has been via the mail and through a bureaucracy. Computer hardware first came into public view in the form of terminals for airline reservation systems, but now they are highly visible in many banks and other institutions.

The list of current and prospective applications of micro-computers that the average person is likely to meet is by now familiar:

- micros in stores and banks – for automated check-out and electronic funds transfer
- micros in the living room – for electronic games, and other entertainment, possibly shopping and voting as well
- micros in the kitchen for controlling ovens and other appliances
- micros in our cars and other vehicles
- micros in hospitals and doctors' offices
- micros in schools for teaching children
- micros at work for word processing systems; process control and telecommunications
- micros attached to our phones
- micros at art galleries and musical performances
- micros everywhere?

Even if not all of these applications come about, micro-computing devices will enter and affect our activities of purchasing, recreation, entertainment, creative expression, health, employment, transportation and communication. This increasing penetration is happening quickly, and though not inevitable, there are strong forces that will keep it going. In short, a new and unique technology will have become part of everyday life for a large segment of the urban population. It promises convenience, efficiency and new capabilities.

The danger in all this is that the benefits will not be available to everyone equally and that the price paid in the long run will be a loss of effective control over the important processes of life. Machines that people do not understand and feel no part of will cut them off from their environment and from each other.

The realization that micro-computers are going to have a major and potentially harmful impact on our society is not, by itself, sufficient reason to becomc actively concerned in trying to affect their development. We need confidence that the technology is somewhat flexible, that it can be changed to some degree, and further that these changes will be reflected back on the society that uses the technology. If, on the other hand, what we do has no effect, then we might as well sit back to enjoy the show and amuse ourselves with our new toys.

> **We shape our tools;**
> **thereafter they shape us.**
> *Marshall McLuhan*

In the early stages of the development of a tool or technology people decide what shape it is actually to take. Often people even deliberately meet to discuss and decide. After that it is we who get shaped by having to conform to the requirements of the tool. At the same time it develops a momentum that is very difficult to change later on even if the results are not totally satisfactory. At the moment micro-computing devices are still crude and many of the details not worked out; the manufacturing and marketing systems not fully developed; patterns of use have not been firmly established; the myths that apply to conventional computers will no longer be accepted but replacements have not yet taken hold; legislative regulations await formulation. In many areas the technology and related structures have not solidified and although to a large extent the future forms have already been determined by prevailing social forces, there is still a short time to influence the course of development. To do this positively and effectively requires a proper understanding of the technology and what it has to offer.

SCHUMACHER'S FOUR CRITERIA

Two contemporary authors have made significant contributions to the understanding of what we should expect from our technologies, and their manifestation as tools, in order that they serve us best. Ivan Illich, writing in *Tools for Conviviality*, calls for tools "which give each person who uses them the greatest opportunity to enrich the environment with the fruits of his or her vision." (page 22), tools "which foster self-realization", (page 25) tools that promote self-reliance, autonomy and harmony between ourselves and with nature. For this to happen, he says, it is essential that control of the tools lie with the individuals and organizations they are intended to serve. E.F. Schumacher has similar objectives, and in *Small is Beautiful* and *Technology and Political Change*, he identifies the *scale* of the technology as the most important factor in determining its social consequences. In general, many present technologies are so massive that the overall effect for people is that they suffer, rather than benefit, from their use. Smaller, more human-scaled enterprises are needed in the formation of a more humane society.

Schumacher suggests four criteria with which to judge technologies and serve to guide their development. These criteria are:

- **small**
- **simple**
- **capital saving**
- **non-violent**

Technologies which satisfy these criteria will be much more likely to be good for us.

At first appearances it would seem that this new micro-computer technology based on Large Scale Integration does satisfy many of Schumacher's criteria and so could have the potential to provide us with tools of great service. These criteria are not really new and in fact are based heavily on common-sense notions of what makes something friendly and easy to get along with. Everyone has had rewarding experiences with tools or institutions and I am sure that they would often find that these criteria were underlying the success of the encounter. Conversely, unsatisfactory experiences with technology can often be traced back to it being huge, complicated, expensive, or violent. By themselves these criteria are not complete and do not guarantee that the effects of a technology will be humane, but they do provide a useful starting point.

Small. People tend to get overwhelmed when they have to approach and deal with things that are much larger than themselves. Big things, whether they are rocks, or ships; companies or governments are generally harder to manage, less flexible, and more resistant to innovation than smaller things. When big things go wrong they tend to go really wrong. Good examples of this are large oil tankers and New York City's electrical power system. It is clear that the physical

size of micro-computers is small. They can be put on your desk, can be carried around, taken over to a friends' house, the whole thing can be seen and be grasped at one time. This is an important departure from the conventional view of computers as large, complicated and somewhat fearsome devices, demanding big buildings to house them, special air conditioning to keep them cool, phalanxes of experts to care for them, and layers of bureaucrats to feed and protect them.

But considerations of size apply not only to the actual hardware that the user sees but also to the back-up technology and institutions required to make effective use of the tools. In many of these respects the "smallness" of micro-computers remains intact.

Manufacture and marketing of most of the hardware components can be done with relatively small firms and without huge factories and machines. In fact, many of the products available to computer amateurs have been introduced by small groups of individuals operating out of "garages". This reflects both the relative newness of the field (there hasn't been enough time for companies to grow really large) and also characteristics of the technology and the particular way in which it has developed. A good illustration of this is found in the "bus" structure that underlies many current micro-computing devices. A "bus" structure is not only an excellent way to organize but has important consequences in other areas. The emergence of the "Altair" or S–100 bus as a de-facto standard has resulted from, and more importantly, allows, small enterprises to create special products that can plug into it. Of course this has depended on the fact that there is no restriction preventing anyone from doing this. It is not hard to imagine that a really large manufacturer (mentioning no names) would either not use such an easily copied and exploited technique or would find artificial means to prevent such "abuse" and preserve its monopoly.

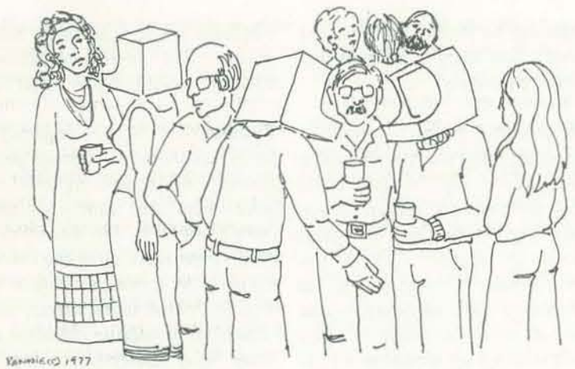
It is a little ironic, but the smallest part of the hardware, the integrated circuits, appear to require the largest enterprises and installations. What this means for instance, is that if you want a chip that the semi-conductor manufacturers don't have available, there isn't very much that can be done unless you have a lot of money. It is very difficult for them to accomodate to small requests and you can't go off and do it yourself. You are almost completely dependent on them. Programmable Read Only Memories (PROMS) are an important step towards independence in certain areas.

On the software end, small groups appear to be able to operate efficiently, perhaps more efficiently than armies of programmers even with very large programming tasks. However, when it comes to marketing a hardware/software combination, size may be a considerable asset in order to enjoy economies of scale and as a protection against competition.

Simple. Simplicity is another virtue, not necessarily related to size, but sharing with "smallness" the tendency to promote autonomous control and self-reliance. If a technology or tool is simple then it is much easier for people to understand it, to use it and modify it for their purposes, to fix it when it goes wrong, in short to control it rather than be controlled by it. Unfortunately micro-computers are simple only in a few, restricted aspects. Most people do not have a great deal of trouble using computer equipment. Small children appear to find it simple enough and although harder to use than most popular appliances (e.g. telephone, oven, etc.) it is certainly much easier than an automobile, once some basic skills and concepts have been mastered. The same goes for the assembly of equipment from kits and the composing of simple programs. However, when involvement goes beyond this level, things become much more complicated. The analysis, design, and programming of major systems requires specialized skills and experience. There do not seem to be programming languages and techniques presently available that allow us to do this sort of thing fairly easily. This is due no doubt in part to the inherent complexity of the process but also because we are still in the infancy of our understanding. The problem is even worse when it comes to the design and manufacture of hardware, particularly integrated circuits. A highly advanced and sophisticated technology is involved, requiring machines of great precision and personnel of rare expertise. With the reliance on experts goes a loss in control and freedom, resulting in a situation akin to that with the automobile. Most people can drive one, put in gas, change tires and sparkplugs, but when it comes to repairs we leave our mobility (and pocketbooks) in the hands of trained auto-mechanics. The prospect of actually making our own vehicle or modifying an existing one is simply out of the question for all but a small fraction of the population – it's just too complicated. The present situation with micro-computers is substantially worse, with the consequence that at least in the foreseeable future their popular use will center mainly around the purchase of plug-in modules. Most people will depend on outside hardware and software suppliers and not be able to rely on themselves or others in their vicinity. Only professionals and avid amateurs will be able to understand and change what is going on. For the most part the guts of our major machines will continue to be generally out of bounds.

Capital saving. The criteria of capital saving means more than just that the products are cheap but applies to all of the financial aspects of a technology. The capital requirements are clearly related to the size and complexity of a technology. Like size and complexity, capital requirements have an important influence on the technology's control and accessibility. Again it is useful to distinguish between the hardware and software aspects and the consumer and production ends of micro-computer technology.

Perhaps the most dramatic and publicized feature of the "micro-computer revolution" is the drastic drop in price of hardware components. This plunge in cost is probably the most important factor contributing to the rise of popular computing and the trend shows every sign of continuing. Small hobby systems are now available for a few hundred dollars, which is in the order of domestic appliances and within the reach of many people. How-



ever, by the time the extra memory, peripheral storage and other goodies have been added to make an interesting system the cost has risen to several thousand dollars, making it a substantial investment and thus out of the range of most individuals and small organizations. With further refinements of technique and the development of mass markets, this will probably remain the case for only a few more years, at least in North America. This contrasts sharply with conventional computer equipment, which is enormously expensive in comparison, and thus only available to already wealthy organizations. At the manufacturing end it appears that great investments of capital are not required *except* in connection with the production and marketing of integrated circuits. Here, expensive and sophisticated equipment is needed. The Canadian Government recently lost $50 million in an unsuccessful attempt to set up its own semi-conductor industry.

Software, to bring the newly inexpensive hardware to life, has not enjoyed the same radical drop in price, mainly because it is a knowledge and labor intensive process and not susceptible to easy automation. To produce high-quality software does not require lots of money to start with but does take a great deal of time and care, making the ultimate cost fairly high. The purchase price can be kept down if the software is general and reliable enough to be attractive to a large number of users. One way to do this is through software exchanges where software can be distributed cheaply. The producer would not stand to earn a great deal of money but would presumably receive some compensation by being able to enjoy other peoples' work at a low cost. If this system is to flourish though, the software must be very reliable, well documented and widely accessible. Publication of software in periodicals such as *People's Computers* is a good example of this process in action at a level of amateur computing. The alternative seems to be stiff copyright regulations maintaining high costs for all but the most widely used packages.

Non-violent. It is very important that a technology that is to find widespread use not harm people nor damage our natural environment. In a direct sense this is certainly true of micro-computer technology. The greatest physical danger to an average user is having one drop on his toe or getting an electric shock. Environmentally, no aspect of computer technology, even with universal application, consumes much in the way of natural resources, demands large amounts of energy, nor produces appreciable pollution. We can say generally that computers "walk lightly on the earth". Less directly however, microcomputers can lead to enormous physical violence since they play an essential role in some of our most destructive and fearsome weapons – notably the "cruise missile" and other "smart bombs". Micro-computing is like other communications technologies in that its effects are usually not directly physical. For instance, the violence that results from television is not due to its physical characteristics but rather to the indirect effects on its viewers over a period of time. With micro-computing, the violence comes from the ways in which it affects how people see themselves and how they relate to others and their environment. Excessive and narrow use of computers seems to promote mechanistic thinking and de-personalization which represent, and result in, forms of violence that, though not overt, are nevertheless significant. Further investigation of this social and psychological violence is vitally necessary if large numbers of people are going to come in regular contact with computers.

OTHER FACTORS

Besides Schumacher's four criteria a number of other important factors should be considered when examining micro-computing's potential to serve as a humane technology in our society.

Fun. Micro-computers are fun! Lots of people, children of all ages, enjoy playing with them. Computers are not just serious business! They have the potential to become great mind toys since they are so flexible and can be adapted to represent such a wide variety of situations (see Ted Nelson's *Computer Lib/Dream Machines*). They are such good toys, in fact, that some people get so fascinated and turned on to the world they are creating and exploring in the machine, that the outside world gets lost. For this reason, excessive exposure to computers should not be allowed to drive out other more physical and social forms of play, particularly in the case of children. Nevertheless the pleasure that can be obtained from micro-computer technology is an important part in making it friendly and socially desirable.

Breakdown of fear and misunderstanding. The widespread use of micro-computers in everyday home and school environments will hopefully have the effect that people will gain a much better understanding of computers. This will tend to lessen the fear of computers based on ignorance and provide a firmer basis for the real, justified fears and thus enable people to deal more effectively with the technology. It could make it much harder for large organizations to hide behind their machines and abuse people by insisting they conform to the expectations of the computer. People will learn that it is not the machinery that is mainly responsible. However, there is a danger that

the understanding may only be superficial and that the loss of fear resulting from familiarity may work against people. Instead, what could happen is that this simply paves the way for easier penetration and acceptance of the more questionable aspects of computer technology, such as Electronic Funds Transfer and other large-scale databank systems.

Redistribution of computing power. The ability to process information quickly and cheaply is an important component in the exercise of power in present society. Until now the mechanisms for doing this have rested almost exclusively in the hands of large institutions while individuals and small organizations have been pretty well left out. The wide-spread availability of small, yet reasonably powerful computers could alter this significantly and help shift real power, along with computing power, toward smaller social units. It is important to realize that computing power is not real power. You must already have some of your own power before a computer can increase it. For a shift to take place not only would the computing power have to be accessible but also the expertise and raw information needed to go with it. It is not clear that the large organizations whose strength would suffer as a result would be willing to relinquish their control over these as well as the other vital components of power. In fact, if a multitude of micro-computing devices act mainly as peripherals in a highly centralized system, then power will be further consolidated in the hands of the massive interests that already dominate society.

Crude, not powerful. Cheap, general purpose micro-computing systems are still in their infancy and are as yet fairly crude and lacking in significant power. This is particularly true of the way in which people have to interact with micros. The graphic quality of output, based as it usually is on a stark, single upper-case type font of poor definition is often much lower than we are accustomed to in other areas. This doesn't have to be the case. Input to computers is almost always through keyboards with very little use being made of devices such as light pens, joy sticks and "mice", which, in many cases, are more appropriate and easier to operate. Programming languages with which we communicate with the machine are also often primitive. Seldom do they offer convenient aids in learning them, are usually intolerant of any "mistake", rudely insisting that they be addressed in strict and peculiar ways, and they do not easily admit to modification to correct their annoying features. In fact, a computer usually has to be treated rather like a spoiled child from whom any desired, out-of-the-ordinary behavior can be obtained only by those who are very familiar with the brat or who love it as a mother and are therefore blind to its shortcomings. This is not to say that there have been no important achievements, for clearly there have been, but we still have a long way to go before most people find micro-computers to be friendly, useful and easy to get along with.

Exclusiveness. Micro-computing is much more accessible to average people than conventional computing, but still has in some ways the same exclusive "cult" feeling about it. For example, informal observation reveals that the overwhelming majority of active participants are male. Why is it that more women are not involved? Is it because machines are inherently more interesting to men than women? Perhaps women receive cues that they are not wanted in the area? The simple fact that it is already so male-dominated may be sufficient to intimidate many women from participating actively.

The sources of discrimination and exclusion are probably deep-seated in our culture and so cannot be removed quickly. However, there are some fairly obvious mechanisms that can be seen at work and dealt with. One of the clearest is the way that jargon is used. Any advanced technology needs specialized language in which to express the notions that are peculiar to it. However, at the same time, and quite unconsciously, it also serves to exclude those who are not initiated into the circle and reinforces membership for those already in. Deliberate avoidance of unnecessary jargon and careful explanations for novices will do much to broaden participation in this important field.

THE CHALLENGE

It is hopefully clear from what has preceded, that micro-computer technology, though suffering from some major drawbacks, has a number of positive features to recommend it for wide-spread, human scale use in society. How the technology is shaped now, in the critical period of its infancy, will have long-term consequences. One group of people who will influence the course of development and who are in a position to exercise some choice over its direction are the computer amateurs. There are two main reasons why this is so. The primary reason is that computer amateurs are the first major group in the general population to have significant computing devices in their

homes. If home computers catch on, the market would be enormous and so major manufacturers will be watching closely to see what happens to get an idea of what will sell. Behavior patterns, in terms of purchase and use of micro-computer systems will establish trends that will continue for some time. Hopefully high standards will be set during this early period when the market is still forming. If people insist on quality systems that are reasonably powerful and self-contained, that give the user flexibility and choice to implement interesting, constructive applications, then the result will be quite different than if people tolerate the introduction of crude pre-packaged systems of narrow imagination and usefulness into their homes.

The second reason that computer amateurs can influence the evolution of micro-computer technology is that they have the time and freedom to explore interesting avenues of approach. Many amateurs have tremendous technical expertise that can be applied to a wide range of applications. They are not bound by having to make money from the end product so they can try many approaches before achieving a really significant result. For instance, an amateur computerist could enjoy spending a great deal of time collaborating with a musician on a music system before coming up with a version that was truly exciting. It could then be made available to many others at very little cost. In other words, computer amateurs represent an important pool of inexpensive research talent and possibilities.

One consequence of the influential role computer amateurs occupy at the present moment is that they have a social responsibility to others to use their position wisely. This responsibility comes from the fact that the actions of computer amateurs could have a significant effect on other people's lives and is independent of whether they actually choose to acknowledge the responsibility. This is one of the reasons that the term "home and personal computing" is more appropriate than "computer hobbyism" for what computer amateurs do, even though many regard it as a hobby. Because it has major social consequences it is not just like any other high technology hobby. "Computer hobbyism" is simply the thin; leading edge of the much larger field of "home and personal computing". The challenge to the amateurs and other pioneers in this field is to accept a share of the responsibility and work actively to ensure that the micro-computing for home and personal use grows in humane and positive directions.

WHAT TO DO

If the argument has been accepted so far, the question now is: "What does one do about it all?" To answer this question directly is very difficult so what I have here are some suggestions for things that can be tried, rather than "answers". The suggestions are tentative and sketchy, at best, but should provide a good starting point.

Research and discussion. One obvious step is to find out more about the whole area of computing and its social implications. Two books that can help in this area are *Computer Power and Human Reason: From Judgment to Calculation* by Joseph Weizenbaum and *The Conquest of Will: Information Processing Human Affairs* by Abbe Mowshowitz. Discussions with others will also serve to expand the base of knowledge and refine viewpoints. Try to keep a broad view of one's activities so that technical considerations are balanced by human concerns; so that people don't get lost in all the bits, bytes, and bauds.

Legislative action. It has often been observed that the law lags far behind technological innovation. This is certainly true for legislative changes required by the increased use of computing devices, particularly in such areas as individual rights to privacy and access to information, the setting of industry standards, and the incorporation of regulatory bodies. Computer amateurs have a special role to play in legislative action because they can speak as independent citizens but with a solid technical foundation. When acting in groups they are in a good position to evaluate and comment on legislative proposals.

Follow Schumacher's four principles. The criteria of smallness, simplicity, capital saving, and non-violence serve as a useful guide in developing both hardware and software systems. In creating new hardware components other people with similar needs will benefit most if the design is kept simple and inexpensive. With software development, concentration on high-quality, well-documented, and transportable programs, though quite difficult to achieve in practice, will likewise result in widest benefit. Programs should treat users well by being polite, by providing learning aids, good error handling and clean interfaces. Increasing the sophistication of programs in this way may require considerable extra effort and contradict to some degree the aim for simplicity, though imaginative approaches are often able to overcome dilemmas of this nature. If a program is worthy of the attention of other people, software exchanges and publications like *People's Computers* are a good way of making it available to them cheaply.

Build on existing cooperative and sharing enterprises. The social and organizational settings in which micro-computer technology is developed and used will define precedents of some lasting effect. If the work is done in an environment of relaxed and open exchange of information by people in small local groups that are in close communication with each other, then the results are likely to be more humane than if it is done in a competitive and secretive atmosphere. Computer clubs, which are sprouting up everywhere, are good starting points because they provide an opportunity for people living in the same area to get together to share experiences, exchange ideas, software and the like. Club newsletters are a valuable means for keeping members in touch with each other, and trading them between clubs expands the communications network even further. Computer amateurs have of course their own novel communications media at hand and so direct computer-to-computer link-ups may prove useful and at least interesting. (see the articles on personal computer networks in the July-August and September-October issues of *People's Computers*).

Work with non-computer people. A very important step is for non-computer people to be involved with the development of micro-computer technology. By non-computer people I mean those whose main motivation is to do something particular, such as draw a picture, make music, or organize and share information. Such people look upon a computer as simply one of perhaps several tools for accomplishing their objective. Computer people on the other hand (to take an extreme view), look for things they can use their computers on. They love their machines and the actual process of using them, rather than the end product. As a consequence these two groups have very different approaches. The computer person can explore wild and fantastic uses and show what the machine can do. Their vision is limited by what they perceive to be the bounds imposed by the machine. Non-computer people will bring them back to earth and supply interesting ideas that go off in quite different directions and which push hard on the limits of what the technology is capable of. The results of the interaction of these two forces should be excitement and conflict. Such interaction will no doubt be difficult at times but is quite vital to the health of micro-computing. Growth by incorporating a variety of perspectives is essential to avoid repeating the mistakes of conventional computing that have resulted from computer people being given too much unbalanced influence.

Computers are too important to be left entirely up to computer people.

There are two important requirements that must be satisfied if this collaboration is going to work. First, it is obvious that the two parties are going to have to be able to communicate effectively and this will require a form of commonly understood language. Each will have to learn a little of the other's terminology and at the same time reduce their own use of special jargon. It will take some time before this gets smoothed out. The second need is for the non-computer person to be able to communicate with the machine in a language that is natural to use and which can express the notions peculiar to the particular application. To start with the computer person will probably act as a go-between but in many cases the lack of immediacy will be unsatisfactory and a special language for direct communication will have to be implemented on the machine. This too can be expected to take much time and effort. In short then, the message to the concerned computer person is to get involved with individuals or small groups (artists, musicians, writers, teachers, business people, cooperatives) that one supports, and work with them on mutually rewarding projects. The going won't be easy but it will certainly be stimulating and contribute valuably to the future positive and beneficial use of computers.

Suggested applications. Hand-in-hand with these non-computer people there are a number of promising applications that are worthy of attention. Some are conventional applications that can be adapted to a smaller scale, while others are fairly innovative. Many have already been tried in one form or another, with varying degrees of success. It should be pointed out that applications are not, by themselves, "humane". They have also to be judged in the context of their actual use. Any computer system must be appropriate to the particular circumstances and be controlled by, and serve the needs of, the people affected by it.

Design. Many small projects involving design (e.g. boats, solar energy, insulation, etc.) require calculations appropriately done with a micro-computer. For example in Ontario a van equipped with a computer goes around to small businesses and helps determine the insulation needs and resulting energy savings.

Process control. A small machine shop may be able to use a micro-computer based process control system to permit the easy set-up of relatively small production runs and the convenient incorporation of custom modifications. The savings in re-tooling costs could allow the shop to remain competitive with larger operations.

Bookkeeping. Small businesses, theatre groups, farmers all need to do a certain amount of bookkeeping and often have neither the time nor inclination to do it themselves nor the money to hire someone else. A simple and cheap micro-computer based accounting system may be just what they need. In Canada, the government provides an inexpensive nationwide accounting service to farmers. There is no reason why this sort of thing cannot be done on a local basis with a small computer.

Aids for the handicapped. Handicapped people, particularly those who are blind or deaf, need to have sensory information converted to a different form before it can be perceived. A small microprocessor unit could be very appropriate. Also handicapped people who cannot use the telephone in the conventional manner would benefit greatly from a scheme for computer communication. Deaf people fall into this category, as do those suffering from diseases such as cerebral palsy.

Art. The potential for application of small computers in the arts is very broad. Computer-based "music boxes", animation, stage-lighting, sound synthesis, graphics, poetry have all enjoyed a measure of success.

Games. Games are fun and, if designed with care, can be educational – not only for the programmer but also for the player. Gaming is probably the area that has received most attention from computer amateurs and there are already plenty of games around. A great number of these are one-person conflict games and there seems little need for more of them. Games that involve several people, that depend on cooperation or negotiation, that teach useful skills or encourage the understanding of important processes would all be worthwhile and provide balance to the scene. Game scenarios could be taken from realistic situations such as ecological systems, government, mathematics, land-use, physics, transportation and so on.

Information/Communication. Many small organizations have filing systems that would be greatly improved through automation, if it was sufficiently cheap. Not only would the routine up-dating chores be much easier, but wider access to the information in the files would be facilitated. Richly cross-indexed directories and catalogues could be produced quickly and more frequently. Mailing lists, in particular, are a prime candidate for this type of application of micro-computers. Exciting possibilities open up when systems of this kind are set up that allow direct access by a whole community of users. People can not only get out previously stored information but put their own in as well. The system thus serves as a memory and communications medium for the community. In fact, the first two prototypes were known as "Community Memory". The original one operated in the San Francisco area and was later "cloned" to Vancouver with the help of the author. Both versions used large time-sharing computers and work is now under way on a stripped down verion (Lee Felsenstein's "Cruncher") that will run on portable micro-computers.

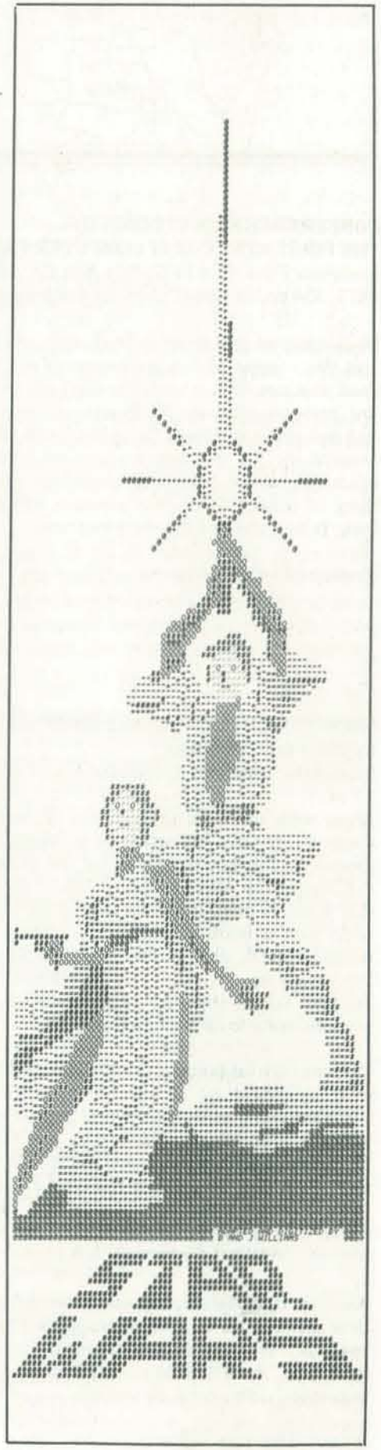
This quick rundown of suggested applications is quite incomplete and unspecific, (for more see Ted Lau's "A Catalog of Liberating Home Computer Concepts." *BYTE* – May 1977) but it does give an idea of some of the things that an individual or group with time, energy, and a small computer on their hands could fruitfully undertake. They all represent promising steps in the development toward a humane and powerful technology.

CONCLUSION

To return to the question posed originally, "If small is beautiful, is micro marvelous?", the answer must be:

Now? "No!",

But perhaps "Yes!", if we make it so.



BIBLIOGRAPHY AND REFERENCES

These are some of the books and articles that relate to the subject of the social relevance of micro-computing.

Ken Colstad and Efrem Lipkin, *Community Memory: A Public Information System,* paper presented at the Computer Science section of an IEEE conference, San Francisco, February 1975.

Erich Fromm, *The Revolution of Hope: Towards a Humanized Technology,* Harper and Row, New York, 1968.

Karl Hess, *'Community Technology', SPARK,* Vol. 4, #2, pp 10-15, Fall 1974.

Ivan Illich, *Tools for Conviviality,* Harper and Row, New York, 1973.

Abbe Mowshowitz, *The Conquest of Will: Information Processing in Human Affairs,* Addison Wesley, Reading, Mass., 1976.

Ted Nelson, *Computer Lib/Dream Machines,* Hugo's Book Service, Chicago, 1974.

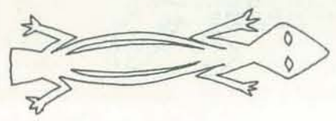
Victor Papanek, *Design for the Real World,* Thames and Hudson, London, 1972.

E.F. Schumacher, *'Technology and Political Change',* Reprinted from *'Resurgence'* in *RAIN,* Vol. III, #3, pp 8-10, Dec. 76.

Joseph Weizenbaum, *Computer Power and Human Reason: From Judgement to Calculation,* Freeman, San Francisco, 1976.

Norbert Wiener, *The Human Use of Human Beings: Cybernetics and Society,* Doubleday, New York, 1954.

Mike Wilber & David Fylstra, *Homebrewery vs. the Software Priesthood, BYTE* Magazine, Oct. 1976.



REVIEWS

CONFERENCE PROCEEDINGS OF
THE FIRST WEST COAST COMPUTER FAIRE
Computer Faire, Box 1579, Palo Alto CA 94302
1977, 334 pp, $12.68 ($13.40 for Californians)

What can you say about a book with one hundred authors? Just this – some of it is good, some of it is not, but it is more good than not. With a book like this I can't make any hard and fast recommendations. What any one person will find good and bad about this book depends a lot on what he or she is interested in, the information they are looking for, and how much they know. I do think almost everyone will find something of interest in it. Find someone with a copy and take a look. It might be just the thing you need.

Reviewed by Eryk Vershen.

MICROCOMPUTER TROUBLESHOOTING MANUAL
by Micro-Info Associates
Micro-Info Associates, Castroville, CA 95012, 20 pp., $5.00

Along with the rapid proliferation of hobbyist-owned home computer systems has arisen an increasing need for a concise general troubleshooting guide for the technical beginner. Unfortunately the *Microcomputer Troubleshooting Guide* is not that book. It does, however, serve as a fairly decent outline of what such a book should include. There are indications of things to check when problems are encountered – CPU clock, I/O, power supply, etc. – but very little on the necessary procedures. Nor is there any mention of systematic diagnostic procedures for isolating a problem.

The most useful function of this manual has been to point up what is needed by its own failure to provide it. A good troubleshooting guide should first provide diagnostic steps for isolating a problem as far as this can be done before just jumping in with test equipment. It should then provide information on how to use that test equipment as specifically applied to microcomputers. Guidelines for memory testing should take up a large portion of any repair manual – especially software guidelines for memory test programs.

Another vital feature, especially for the amateur, would be clear instructions on how to recognize when he is in over his head. A liberal use of diagrams and examples would help a great deal. The definitive troubleshooting guide for microcomputers has, sadly, yet to be written.

Reviewed by Tom Williams.

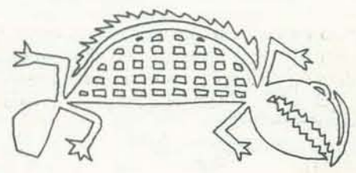
A DRAGON HUNTER'S BOX
by H. A. Cohen
Available for $10.00 (U.S.) from Hanging Lake Books, POB 157, Warrandyte, Victoria 3083, Australia; allow 10 weeks for delivery.

This is actually a box about the size of a small book containing about 75 unbound pages of hand lettered text and delightful drawings illustrating the 'dragons'. Dragons in this box are puzzles that require careful use of high school applied mathematics in their solution. Each dragon is presented as an imaginative miniproject together with hints on their solution.

Some of the projects succeed remarkably well in being challenging and result in intriguing hunts. Unfortunately the presentation occasionally becomes unnecessarily ornate and arcane rather than imaginative. Further, the actual number of projects (about 30) is relatively small and one is left wishing the author could have provided more. Though attractively presented, the lack of a binding or at least a page numbering scheme, means a reader tackling the box for the first time may not, for example, find the introduction until (s)he is half way through the problems, an annoying confusion.

These criticisms aside, the dragons should provide fun for those who enjoy tackling mathematical puzzles.

Reviewed by Les Cotrell.



COMPUTER CAREERS:
PLANNING, PREREQUISITES, POTENTIAL
by John Maniotes and James S. Quasney
Hayden, 1974. 180 p., $4.95.

This is an excellent book, and should prove invaluable to anyone planning a computer career and also to high school and college counselors; teachers of mathematics, science and business courses; librarians; parents; and anyone else interested in the computer revolution. Unlike some books on computers, *Computer Careers* outlines not only the opportunities in the field but the problems and pitfalls as well.

Chapter 1 provides an overview of the career opportunities in computing. The authors point out that the number of professional programmers is expected to increase from 200,000 in 1972 to 400,000 by 1980. They also note that women have excellent career opportunities in computer-related fields, even at the highest salary levels, but caution the reader that the rewards are not obtained without sacrifice, hard work, and advanced education.

Chapter 2 explains briefly what computers do, what their major components are, and what a computer program is. This chapter also contains one of the very few errors in the book; on page 27, a Burroughs alphanumeric display is incorrectly identified in a photograph as a CRT.

Chapter 3 discusses the categories and job titles commonly used in computer-related jobs, but the reader is warned that job titles currently in use are often confusing and misleading. Job categories are further defined in Appendix F. Basic educational requirements and salary ranges are given for various jobs.

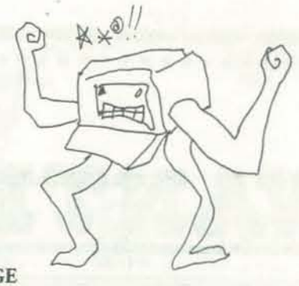
Educational requirements and how to meet them are covered in considerable detail in Chapter 4. The book emphasizes the fact that the requirements for a career in computer science are quite different from those for an EDP-related career, and typical curricula for both are outlined in Appendices G, H, and I.

Some practical ways of financing a computer-EDP education are suggested in Chapter 5, including tips on obtaining scholarships and student loans.

Chapter 6 deals with the techniques of job hunting. This chapter alone would be well worth the price of the book. Here the authors map out a detailed strategy for finding a computer-related job, beginning with the theory and practice of writing a resume. As stated in the book, few job seekers appreciate the importance of a good resume. A resume is perhaps the most important single document any of us will ever write, with the possible exception of a last will and testament. Preparation for job interviews and aptitude tests is also covered.

Professionalism is discussed in Chapter 7, and a comprehensive directory of computer-EDP schools is provided in Chapter 8. A wealth of supplemental information is presented in the appendices, and each chapter includes a list of books and articles for additional reading.

Reviewed by Jim Day.



COMPUTER RAGE
a board game from *Creative Computing*
PO Box 789-M, Morristown, NJ 07960, $8.95

Computer Rage is a simple race game couched in computer lingo. Its essence can be described in one word – Luck.

The parts of the game are: a board with a long, simplistic, colorful flowchart on it; a deck of thirty-eight Interrupt cards; four sets of three colored pieces ('programs'); and three 'binary' dice. A binary die has a zero on three sides and a one on the other three sides.

The object of the game is to get all three of your pieces from the input symbols of the board to the output symbols before anyone else. There are, of course, hazards. You can land on a square containing an interrupt symbol and draw a bad interrupt card; or the dice might give you a long path at a decision point.

At two places on the board (after the input section and before the output section) there are a pair of paths called 'channels'. Only one piece can be in one channel at a time. This creates the tactic of 'blocking a channel pair'. The tactic can give the first players (in a four player game) an advantage.

We tried out the game with 4 players. Initially there were problems in remembering which order to read the dice (red - blue - green) in part because no reminder was printed on the board. The board, playing tokens, and dice are all multicolored and it was sometimes hard to locate the dice in the profusion of shapes and colors.

Comments about the game varied. The only 'computer' element of the game that was commented on was the need for learning binary notation to facilitate reading the dice. Said loser Bill 'No challenge.' Said loser Margo 'The game needs more variables – I think you should be able to build hotels. But then I'm a builder, not a computer person.' Said winner Lisa (age 12) 'It was a good game – the right combination of luck and strategy. I liked it, but it was basically just rolling dice and moving pieces, not very different from other board games. And I think it should cost $3 to $3.50, not $9.'

Reviewed by Utter Chaos and friends. □

WANTED

All readers of PC to be on the lookout for a special beast

Hi! PCC is doing a second computer games book. We want this to be the best book yet, so we're asking your help. We don't claim to know all that much about making games that appeal to people. We need to know what you want.

What kinds of games do you like? What do you like and dislike about current computer games? What do you want to see in a computer games book? What computer language or dialect are you using? What are its limitations? Maybe you have a game program you'd like to see published?

We'd like to hear about it. Send your comments, suggestions, criticisms, and programs (with a SASE if you wish your programs returned) to:

PCC
Attn: Eryk
Box E
Menlo Park CA 94025

DATA HANDLER
Sample Program

The following program illustrates multiplication by successive addition. It is an answer to the exercise posed in the Data Handler article on page 12.

PROGRAM FOR MULTIPLICATION
BY SUCCESSIVE ADDITION

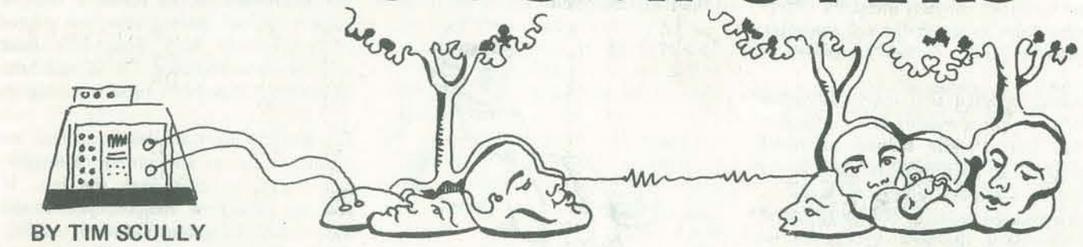
LABEL	LOC	INST	COMMENTS
	FC00	A2	Load X register with
	FC01		First number.
GO	FC02	AD	LDA
	FC03	50	
	FC04	FC	
	FC05	D8	CLD
	FC06	18	CLC
	FC07	6D	ADC
	FC08	61	
	FC09	FC	
	FC0A	8D	STA
	FC0B	61	
	FC0C	FC	
	FC0D	B0	BCS HIORD
	FC0E	06	
COUNT	FC0F	CA	DEX
	FC10	D0	BNE GO
	FC11	F0	
	FC12	4C	JMP FINIS
	FC13	21	
	FC14	FC	
HIORD	FC15	AD	LDA
	FC16	60	
	FC17	FC	
	FC18	18	CLC
	FC19	69	ADC immediate
	FC1A	01	
	FC1B	8D	STA
	FC1C	60	
	FC1D	FC	
	FC1E	4C	JMP COUNT
	FC1F	0F	
	FC20	FC	
FINIS	FC21	4C	JMP FINIS
	FC22	21	
	FC23	FC	

The X register holds the multiplier.
FC50 is loaded with the multiplicand.
FC60 holds the high order product.
FC61 holds the low order product.

□

Part II

BIOFEEDBACK & MICROCOMPUTERS



BY TIM SCULLY

Although this series is titled 'Biofeedback and Microcomputers', so far we've mostly discussed the use of microcomputers for physiological data analysis. We'll get to computerized biofeedback soon, but it will be worthwhile to also talk a little more about physiological data analysis by computer: it may be the key to decoding the body's language and opening up new modes of intra- and inter-personal communications and new possibilities for exploring inner space,

SENTICS:
EMOTIONS IN A FINGERTIP

Manfred Clynes,[1] a musician and biocybernetic researcher, has approached the problem of decoding the body's language from a unique direction. He asked experimental subjects, at a signal from his computer, to express various emotions by movements of one fingertip, while his computer recorded the fingertip's responses. A strain gauge was used to convert variations in fingertip pressure into a varying electrical signal which an A/D converter communicated to the computer.

The computer collected samples of data from the strain gauge at regular intervals. If one recording of these repeated samples of strain gauge data were graphed by the computer, the result would be a picture of fingertip pressure as it varies with time. As in the brainwave experiments described in the last installment, several recordings were made for each emotion, and the resulting patterns were averaged together. This averaging process smoothed out random variations in fingertip pressure and produced a pattern typical of one emotion for the person whose fingertip was tested.

The results of this process were a set of curves of fingertip pressure versus time for emotions such as love, hate, joy and sadness. The purpose of Clynes' experiment was to see if the body has a universal language for expressing emotions, a language independent of cultural and linguistic differences. Clynes traveled extensively and tested subjects from many different cultures: he found that the patterns of fingertip pressure for these emotions were distinctive and cross-cultural.

Clynes called these patterns 'essentic forms', and he reported that the essentic form for anger is the same for Mexican, Japanese, American and Balinese fingertips: a brief sharp pulse of pressure. The pattern for love is universally a gentle curve.

In more recent research,[2] Clynes has collected fingertip responses to musical works by different composers and has once again found patterns of response typical of each composer. He even found that the essentic forms were similar for different subjects and between different works by the same composer.

To explore essentic patterns with your computer, you need to be able to communicate fingertip pressure data to it. A strain gauge will do the job, but may be a little expensive. A material called 'Dynacon A' (Dynacon Industries, 14 Bisset Drive, West Milford, N.J. 07480) might do the same job with less cost. In any case, a simple 8 bit A/D converter (now under $10!) will suffice to complete the interface to your computer. If I can get it together, a future article may give an exact circuit diagram and a program for producing essentic forms with a Poly video system.

A MIRROR FOR EMOTIONS

Gary Schwartz,[3] working at Harvard University, has done some very interesting work in uncovering patterns of muscle tension in the facial region which are related to emotions. He learned that there are patterns of muscle tension which are too subtle to produce visible

facial expressions but which correlate well with emotions felt.

Schwartz used an electromyograph (EMG) to measure muscle tension in four areas of the face: the frontalis muscle in the forehead, the corrugator muscle near the eyebrows, the masseter which operates the jaws and the depressor muscle which circles around the lips and down the chin. He measured and averaged EMG levels in these muscles over 30 second epochs. He compared average resting tension levels with the readings obtained when his subjects imagined different emotions, in much the same manner as Clynes' subjects.

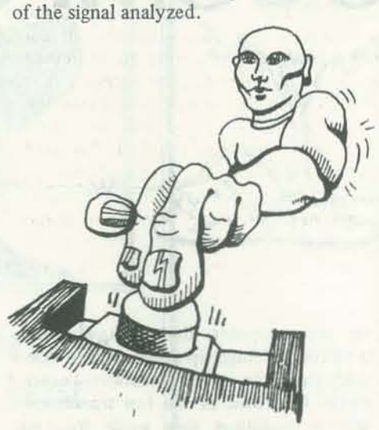
Schwartz reported that happy thoughts correlated with unusually low corrugator tension levels, while sadness produced unusually high corrugator tension. Anger tended to produce unusually high depressor muscle tension. Schwartz found little change in the masseter and frontalis muscles for these three emotions, so it appears that a two channel EMG system may be enough to identify them.

Can you imagine a two channel EMG system interfaced to your computer with a graphic display showing the facial expression that corresponds to your hidden emotions? Such a mirror for emotions is under development for use in counseling, and a future article may describe the hardware and software for it in more detail. The system will use Aquarius Electronics EMG amplifiers and S-100 compatible interface, along with a Poly video card.

INSIGHT: GETTING A BRAINWAVE

Norman Don [4] used a computer to search for brainwave patterns that might identify moments of insight. To understand his experiment, it will help to review a bit of the history of electroencephalography (the measurement of brainwaves). When Hans Berger first recorded human brainwaves in 1929, he attempted to classify and categorize the constantly changing waveforms which he observed. He used amplitude and frequency differences to establish rough categories for types of brainwaves. The use of amplitude and frequency domain features to classify brainwaves is still popular, and you may have heard of alpha waves (8-13 Hz and fairly high amplitude), beta waves (13-30 Hz and low amplitude), theta waves (4-8 Hz and higher amplitude) and delta waves (less than 4 Hz and often very high amplitude). Usually a mixture of different brainwave types can be observed at each scalp location.

The fast Fourier transform (FFT) is a mathematical technique, developed for digital computers,[5] which is designed to decompose a waveform into different frequency bands and determine the intensity (squared amplitude) of the signal in each frequency band. The result of an FFT calculation is a power spectrum of the signal analyzed.



Don used FFT to analyze a single channel of brainwaves in his research. His subjects were graduate students in psychology who were practicing an introspective technique called 'focusing' while tape recording a running commentary of events in consciousness along with their EEG data. Each subject reviewed his tape recorded commentary of his session and identified those times, if any, during which insight or 'ah-ha' experiences took place. Don then used a computer to do FFT analysis of the tape recorded EEG data. He broke the long tape into short time segments called epochs, each 2.56 seconds long, and instructed his computer to calculate in FFT power spectrum for each epoch. A typical session consisted of 1024 epochs.

Don wanted to test his hypothesis that insight experiences would be correlated with unusually high peaks in the alpha band and in its two subharmonics in theta and delta. He wrote a computer program to sort out those which matched this general pattern – he was right: the insight experience did happen during these epochs.

The next step will be to build a biofeedback system, using a computer, to train people to produce this three-peaked brainwave pattern. Then it will be possible to find out if training for this pattern of brainwaves helps promote more insight experiences.

MICROCOMPUTERS AND DESENSITIZATION

Desensitization is a technique used to help people lessen attachments and get over phobias. For example, a person might use it to lessen a fear of heights. Before getting into how microcomputers can help here, let's look at how desensitization works and how biofeedback has been used in doing it.

The simplest form of desensitization involves setting up a hierarchy of images, each more arousing than the last. If you are afraid of heights, you might start out imagining yourself safely in your favorite chair in your living room, then walking outdoors and up a few steps onto a low platform with a railing, then imagine removing the railing and then begin raising the platform up slowly. If imagining this series of images is all you did, the result would probably be just discomfort or even fear. The trick to desensitization is to stop as soon as fear or discomfort is felt, and back up to an earlier, less threatening image until you can relax again. Then try moving forward to more arousing images until fear begins again. If this process is continued, you find that it is possible to make more and more progress through the series of images as you practice more. You are becoming desensitized to images of heights.

One simple form of biofeedback desensitization was developed by Byron Allen and Ken Lebow[6]: a slide projector controlled by galvanic skin response (GSR). You may recall from our last article that a GSR is a brief drop in the electrical resistance of your skin. Byron and Ken's projector advanced to a new slide at regular intervals as long as no GSR response happened. But if a GSR occurred, indicating autonomic nervous system arousal, then the projector would stop advancing and back up to earlier slides. Once the GSR was over, the projector would resume advancing.

Byron and Ken worked at Lompoc Federal Correctional Institution and used their system to help men who suffered from an uncontrollable desire for heroin. Their slides were photos of heroin being bought and used. Their clients initially showed uncontrollable GSR responses to every such slide, but by working with the desensitization system they were able to learn to control this response.

Byron and Ken found that some of their clients learned to control their GSR responses while watching the slides, but still experienced strong feelings. These men had learned to suppress one physiological response to the images without learning to control their feelings. These same men tended to get cold hands while watching the slides. You may recall that stress causes the body to withdraw blood from the hands and feet in preparation for 'fight or flight', and that this causes cold hands.

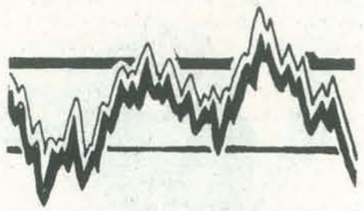
Aquarius Electronics now sells a slide projector controller just like Byron and Ken's. After designing that system, I later built an improved system which used both skin temperature and GSR to control the slide projector. Slides advanced at regular intervals only if hands were warm and no GSR response was detected. Cold hands would cause the projector to stop advancing and 'freeze' until the hands warmed up again. A GSR response caused the projector to back up three slides and then stop. This system worked out well in a pilot study early this year with outpatients in the chemical dependence program at Gladman Memorial Hospital in Oakland, California.

Slides are pretty effective stimuli for this kind of training, but they are not as powerful stimuli as real life situations. Mild habits or phobias, such as cigarette smoking, may not respond strongly enough to slides for a biofeedback desensitization system to work. Combined audio and visual stimulation might be more effective in many cases.

I recently saw an ad for a microcomputer controlled cassette tape drive which has one digital data track parallel with a second audio track on which voice can be recorded. It would be very simple to use a computer controlled cassette deck and slide projector system to do combined audio-visual desensitization training. The physiological data would be fed into the computer, which could compare these data with program-controlled threshold values and decide when to advance or reverse the slides and audio. Each slide could have 10 or 15 seconds of audio associated with it which would begin playing at the same instant the slide appears on the screen.

Even without the audio feature, a computer-controlled desensitization system makes a lot of sense. The computer can be programmed to 'shape' the task so that it is not too difficult at first and so it isn't too easy as learning progresses. This is done by readjusting the thresholds which control the slide projector. The computer can also easily readjust the time each slide is displayed and it can keep good records of training purposes.

We'll be talking more about computer 'shaping' of biofeedback training in part III of this series.



REFERENCES

1. Clynes, Manfred *Annals of the New York Academy of Sciences* 220:57-131 (1973).
2. Clynes, Manfred *Sentics; the touch of emotions* Anchor-Doubleday, New York (1977).
3. Schwartz, G. Biofeedback, self-regulation, and patterning of physiological processes. *American Scientist* 63:314-324 (1975) reprinted in: T.X. Barber, et al *Biofeedback and Self-control 1975/76* Aldine Publishing, Chicago (1976).
4. Don, Norman S. The transformation of conscious experience and its EEG correlates. *Journal of Altered States of Consciousness,* Winter, (1978).
5. Cooley, J., Lewis, P., & Welch, P. The fast Fourier transform and its application to time series analysis. In K. Enslein, A. Ralston & H. Wilf *Statistical Methods for Digital Computers* (Vol 3 of Mathematical Methods for Digital Computers) Wiley, New York (1977).
6. Lebow, K. & Allen, B. *Finding a Cure for Treatment,* Lompoc FCI DAP Program (1974). □

CORRECTIONS FOR OUR SEPT-OCT ISSUE

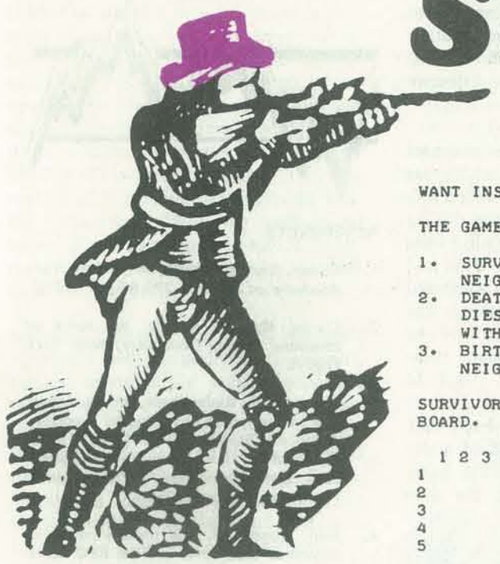
- Utter Chaos provided the photographs in the PET article on pages 22, 26, and 27 of the Sept-Oct issue.
- In the article 'Computer Networking' by Larry Tesler, the next to last paragraph on page 17 should read as follows:

  Let us look at each of these cases. After packet 3 is lost, retransmission (packet 3R) restores order. After ack 4 is lost, retransmission causes D to receive a duplicate packet (packet 4R), which it must acknowledge (ack 4R) so that S will not continue to retransmit indefinitely. After ack 5 is delayed, not only does D receive a duplicate packet (packet 5R), but S receives a duplicate ack (ack 5R), which it simply ignores.

TRS-80

Radio Shack is now a real computer company – you can tell because they're delivering their systems late. We still hope to have a system for review in time for the January-February press deadline. Meanwhile, a few bits of information: delivery of systems began August 15; it's confidential just how many systems have been delivered. Both the 4K and 16K systems may be ordered with a 6 – 8 week delivery time. The first TRS-80s were scheduled to reach Radio Shack Stores in October, to inaugurate the opening of the first Tandy Computer Store in Fort Worth.

SURVIVOR



BY MAC OGLESBY

In 1970, British mathematician John Conway of Cambridge University created 'Life' as a board game which starts with a simple pattern of cells and changes according to the 'genetic laws' listed in the Survivor instructions shown at the right. These laws were chosen to make the behavior of the cell population unpredictable–some patterns die out, others seemingly grow without limit, others evolve into stable patterns. Some patterns, such as 'gliders,' appear to travel across the board. Some of the commonest stable forms are illustrated in this article.

Mac Oglesby's Survivor turns Life into a 2-person game written in Dartmouth BASIC for use on a teletypewriter. Normally when players specify the row and column of a live piece the move is hidden by overprinting, but the paper has been advanced in the sample run so that you can see the moves.

Readers interested in a more detailed mathematical discussion of Life should refer to Scientific American, *Oct. 1970 (pages 120-123), Nov. 1970 (page 118), Jan. 1971 (pages 105, 106, 108) and Feb. 1971 (pages 112-117).*

```
WANT INSTRUCTIONS FOR SURVIVOR? YES

THE GAME OF SURVIVOR IS BASED ON JOHN CONWAY'S GAME OF LIFE.

1.  SURVIVALS---EVERY PIECE WITH EITHER TWO OR THREE
    NEIGHBORS SURVIVES FOR THE NEXT GENERATION.
2.  DEATHS---EACH PIECE HAVING FOUR OR MORE NEIGHBORS
    DIES (DISAPPEARS) FROM OVERPOPULATION.  EACH PIECE
    WITH FEWER THAN TWO NEIGHBORS DIES FROM ISOLATION.
3.  BIRTHS---EACH EMPTY CELL ADJACENT TO EXACTLY THREE
    NEIGHBORS--NO MORE, NO FEWER--IS A BIRTH CELL.

SURVIVOR IS FOR TWO PLAYERS.  THE GAME IS PLAYED ON A 5X5
BOARD.

  1 2 3 4 5
1
2
3
4
5

EACH PLAYER HAS A SYMBOL (* OR #) TO SHOW THE LOCATION OF
HIS LIVE PIECES.

THE SYMBOLS (* AND #) ARE REGARDED AS THE SAME EXCEPT WHEN
DECIDING ON THE OWNERSHIP OF A NEW PIECE.  AN EMPTY CELL
HAVING TWO # AND ONE * FOR NEIGHBORS WILL GENERATE A #.
THAT IS, THE NEW PIECE WILL BELONG TO THE PLAYER WHO HAS
THE MAJORITY OF THE THREE ADJACENT PIECES.

TO BEGIN, EACH PLAYER POSITIONS THREE LIVE PIECES ON THE
BOARD BY TYPING IN THE COORDINATES (ROW,COLUMN) OF THE
CHOSEN CELLS.  A COMMA MUST SEPARATE THE TWO NUMBERS.

AFTER EACH GENERATION HAS BEEN PRINTED OUT, EACH PLAYER IN
TURN PLACES ONE PIECE ON THE BOARD.  IF BOTH PLAYERS CHOOSE
THE SAME LOCATION, THAT CELL IS LEFT EMPTY.

THE OBJECT IS TO SURVIVE.  THE GAME ENDS WHEN ONE OF THE
PLAYERS HAS NO LIVE PIECES.

PLAYER [*] --- ENTER 3 LIVE PIECES.
ROW,COL.
?
  1,1
ROW,COL.
?
  2,2
ROW,COL.
?
  2,3

PLAYER [#] --- ENTER 3 LIVE PIECES.
ROW,COL.
?
  3,3
ROW,COL.
?
  4,4
ROW,COL.
?
  2,3
THAT CELL IS OCCUPIED!
INPUT IGNORED.  PLEASE TRY AGAIN...
```

Normally the moves are hidden by the overprinting. Mac advanced the paper so the moves would show.

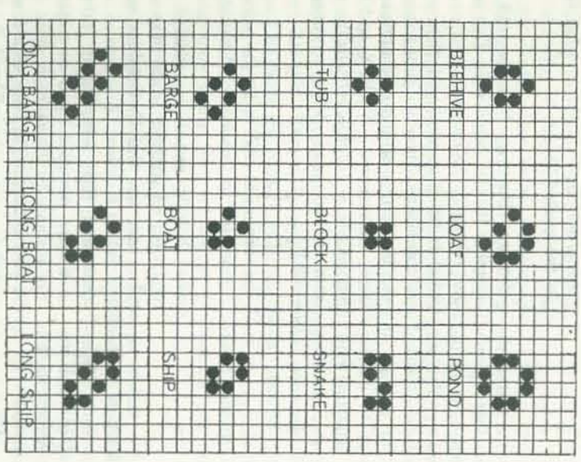

```
ROW,COL.
?
  3,4

HERE'S THE BOARD AT THE START:
  1 2 3 4 5
1 *
2   * *
3     # #
4       #
5
```

Paper-saving format.

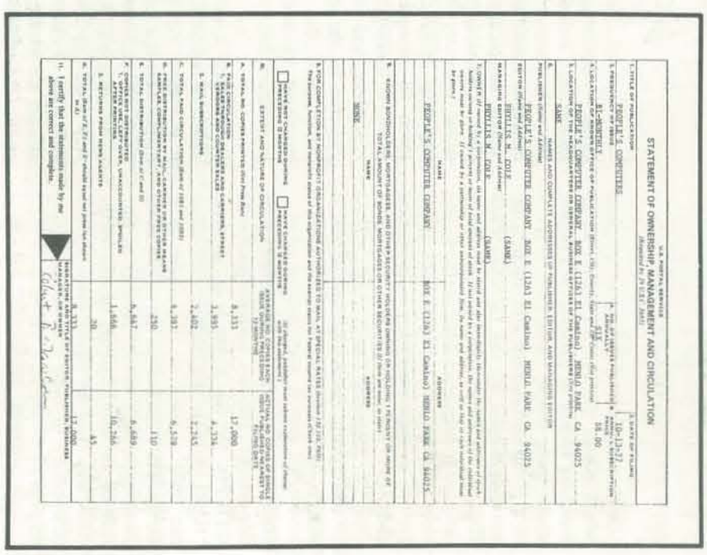
```
AFTER GENERATION 1:
  1 2 3 4 5
1   *
2   * * #
3   *   #
4     # #
5

EACH PLAYER NOW ADDS ONE PIECE.
PLAYER [*]
ROW,COL.
?
  1,4
PLAYER [#]
ROW,COL.
?
  5,1

AFTER GENERATION 2:
  1 2 3 4 5
1   *   *
2 * *   # #
3   *     #
4   # # #
5

PLAYER [*]
ROW,COL.
?
  3,1
PLAYER [#]
ROW,COL.
?
  1,1

AFTER GENERATION 3:
  1 2 3 4 5
1 # *   * #
2       # #
3         #
4 * # # #
5     #

PLAYER [*]
ROW,COL.
?
  2,3
PLAYER [#]
ROW,COL.
?
  2,3
YOU BOTH PICKED THE SAME CELL, SO IT STAYS EMPTY.

AFTER GENERATION 4:
  1 2 3 4 5
1     * * #
2     *
3   #     #
4   # # #
5     # #

PLAYER [*]
ROW,COL.
?
  1,1
PLAYER [#]
ROW,COL.
?
  1,2

AFTER GENERATION 5:
  1 2 3 4 5
1   # * *
2 #       #
3   #
4   #       #
5   #   #

PLAYER [*]
ROW,COL.
?
  3,4
PLAYER [#]
ROW,COL.
?
  1,5

AFTER GENERATION 6:
  1 2 3 4 5
1   # * * #
2 #       #
3 # # # * #
4 # #   # #
5     #

PLAYER [*]
ROW,COL.
?
  1,3
THAT CELL IS OCCUPIED!
INPUT IGNORED.  PLEASE TRY AGAIN...
ROW,COL.
?
  2,3
PLAYER [#]
ROW,COL.
?
  2,4

AFTER GENERATION 7:
  1 2 3 4 5
1   #     #
2 #
3
4 #       #
5   # # #

*** PLAYER [#] IS THE SURVIVOR!

TYPE  RUN  TO PLAY AGAIN.
```

survivor

```
100 ' NAME:  SURVIVOR
110 '
120 ' BY:  MAC OGLESBY  07/02/76
130 '
140 ' DESCRIPTION:  USING THE RULES OF JOHN CONWAY'S GAME OF LIFE
150 '    (SEE "SCIENTIFIC AMERICAN", OCTOBER, 1970 AND FEBRUARY, 1971),
160 '    TWO GROUPS OF LIVE PIECES STRUGGLE TO SURVIVE ON A 5 X 5
170 '    BOARD.
180 '
190 ' INSTRUCTIONS:  TYPE "RUN" FOR COMPLETE INSTRUCTIONS.
200 '
210 ' REMARKS:  THIS PROGRAM IS AN EXTENSIVE REVISION OF BRIAN
220 '    WYVILL'S PROGRAM LIFE-2 WHICH IS LISTED IN "101 BASIC COMPUTER
230 '    GAMES" BY DAVID AHL.
240 '
250 ' RELATED FILES:
260 '      BASICLIB***:TEXTSUB ELICITS AN UPPERCASE YES OR NO RESPONSE.
270 '
280 ' LANGUAGE:  BASIC
340 '
350 '
1000 LIBRARY "BASICLIB***:TEXTSUB"
1010
1020 DIM B(18)
1030
1040 FOR J=1 TO 18                      'B() STORES VALUES WHICH DETERMINE
1050    READ B(J)                       '   WHETHER A CELL WHOSE VALUE IS
1060 NEXT J                             '   STORED IN D(,) BECOMES * OR #
1070    DATA 3,102,103,120,130,121,112,111,12
1080    DATA 21,30,1020,1030,1011,1021,1003,1002,1012
1090
1100 FOR J=1 TO 8
1110    READ R(J),C(J)                  'CHANGES TO ROW,COL. NECESSARY TO
1120 NEXT J                             '   EXAMINE THE 8 ADJACENT CELLS
1130    DATA -1,-1,-1,0,-1,1,0,-1,0,1,1,-1,1,0,1,1
1140
1150 LET P$(1)="(*)"                    'PLAYERS' SYMBOLS
1160 LET P$(2)="(#)"
1170
1180 PRINT "WANT INSTRUCTIONS FOR SURVIVOR";
1190 CALL "YES-NO":A$                   'ELICITS AN UPPERCASE YES OR NO
1200 IF A$="NO" THEN 1240
1210    GOSUB 2460
1220
1230    'EACH PLAYER STARTS WITH 3 PIECES
1240 FOR J=1 TO 2
1250    PRINT
1260    PRINT "PLAYER ";P$(J);" --- ENTER 3 LIVE PIECES."
1270    FOR K=1 TO 3
1280       GOSUB 1920
1290       LET D(R,C)=3+(J-1)*27        'STORE 3 FOR A *, 30 FOR A #
1300    NEXT K
1310 NEXT J
1320
1330 PRINT
1340 PRINT "HERE'S THE BOARD AT THE START:"
1350 GOSUB 2110
1360
1370
1380
1390 'EACH CELL AFFECTS ITS 8 NEIGHBORS
1400 FOR J=1 TO 5
1410    FOR K=1 TO 5
1420       IF D(J,K)<100 THEN 1490      'D(,) NOW STORES 100 FOR A *,
1430          LET E=1
1440          IF D(J,K)<1000 THEN 1460  '  AND 1000 FOR A #
1450             LET E=10
1460          FOR L=1 TO 8
1470             LET D(J+R(L),K+C(L))=D(J+R(L),K+C(L))+E
1480          NEXT L
1490    NEXT K
1500 NEXT J
1510
1520 LET G=G+1
1530 PRINT "AFTER GENERATION ";STR$(G);":"
1540 GOSUB 2110
1550
1560 'GAME OVER?
1570 IF M1*M2<>0 THEN 1690
1580    IF M1+M2<>0 THEN 1610
1590       PRINT "*** A DRAW!!  THERE ARE NO SURVIVORS!"
1600       GOTO 1640
1610    IF M1=0 THEN 1630
1620       LET M1=-1
1630    PRINT "*** PLAYER ";P$(M1+2);" IS THE SURVIVOR!"
1640    PRINT
1650    PRINT "TYPE  RUN  TO PLAY AGAIN."
1660    STOP
1670
1680    'EACH PLAYER PLACES ANOTHER PIECE
1690 IF G>1 THEN 1710
1700    PRINT "EACH PLAYER NOW ADDS ONE PIECE."
1710 FOR J=1 TO 2
1720    PRINT "PLAYER ";P$(J)
1730    GOSUB 1920
1740    IF J=2 THEN 1770
1750       LET E=5*R+C
1760       GOTO 1840
1770    IF E=5*R+C THEN 1830            'CHECK FOR DUPLICATE COORDINATES
1780       LET D(R,C)=1000
1790       LET R=INT(E/5)
1800       LET C=E-5*R
1810       LET D(R,C)=100
1820       GOTO 1840
1830    PRINT "YOU BOTH PICKED THE SAME CELL, SO IT STAYS EMPTY."
1840 NEXT J
1850 PRINT
1860
1870 GOTO 1400
1880
1890
1900
1910 'THE PLAYERS PLACE THEIR PIECES
1920 PRINT "ROW,COL."
1930 PRINT "XXXXXX";CHR$(13);"$$$$$$";CHR$(13);"BBBBBB";CHR$(13);
1940 LINPUT A$
1950 IF LEN(A$)<>3 THEN 2060
1960    CHANGE A$ TO A
1970    IF (53-A(1))*(A(1)-49)<0 THEN 2060
1980       IF (53-A(3))*(A(3)-49)<0 THEN 2060
1990          IF A(2)<>44 THEN 2060
2000             LET R=A(1)-48
2010             LET C=A(3)-48
2020             IF D(R,C)<>0 THEN 2040
2030                RETURN
2040             PRINT "THAT CELL IS OCCUPIED!"
2050             GOTO 2070
2060 PRINT "YOU MUST TYPE 2 DIGITS (1-5) SEPARATED BY A COMMA."
2070 PRINT "INPUT IGNORED.  PLEASE TRY AGAIN..."
2080 GOTO 1920
2090
2100 'UPDATE AND PRINT THE BOARD
2110 LET M1=M2=0                        'RESET PLAYER FLAGS
2120 PRINT "  1 2 3 4 5"
2130 FOR J=1 TO 5
2140    PRINT STR$(J);" ";
2150    FOR K=5 TO 1 STEP -1            'IGNORE TRAILING SPACES
2160       IF D(J,K)<3 THEN 2380
2170          FOR K1=1 TO 18
2180             IF D(J,K)<>B(K1) THEN 2370
2190                FOR L=1 TO K
2200                   IF D(J,L)<3 THEN 2330
2210                      FOR M=1 TO 18
2220                         IF D(J,L)<>B(M) THEN 2320
2230                            IF M>9 THEN 2280
2240                               PRINT "* ";
2250                               LET M1=1  'THE *'S SURVIVE
2260                               LET D(J,L)=100
2270                               GOTO 2350
2280                            PRINT "# ";
2290                            LET M2=1  'THE #'S SURVIVE
2300                            LET D(J,L)=1000
2310                            GOTO 2350
2320                      NEXT M
2330                   PRINT "  ";
2340                   LET D(J,L)=0
2350                NEXT L
2360                GOTO 2400
2370             NEXT K1
2380          LET D(J,K)=0
2390       NEXT K
2400       PRINT
2410 NEXT J
2420 PRINT
2430 RETURN
2440
2450 'INSTRUCTIONS
2460 PRINT
2470 PRINT "THE GAME OF SURVIVOR IS BASED ON JOHN CONWAY'S GAME OF LIFE."
2480 PRINT
2490 PRINT "1.  SURVIVALS---EVERY PIECE WITH EITHER TWO OR THREE"
2500 PRINT "    NEIGHBORS SURVIVES FOR THE NEXT GENERATION."
2510 PRINT "2.  DEATHS---EACH PIECE HAVING FOUR OR MORE NEIGHBORS"
2520 PRINT "    DIES (DISAPPEARS) FROM OVERPOPULATION.  EACH PIECE"
2530 PRINT "    WITH FEWER THAN TWO NEIGHBORS DIES FROM ISOLATION."
2540 PRINT "3.  BIRTHS---EACH EMPTY CELL ADJACENT TO EXACTLY THREE"
2550 PRINT "    NEIGHBORS--NO MORE, NO FEWER--IS A BIRTH CELL."
2560 PRINT
2570 PRINT "SURVIVOR IS FOR TWO PLAYERS.  THE GAME IS PLAYED ON A 5X5"
2580 PRINT "BOARD."
2590 PRINT
2600 GOSUB 2110
2610 PRINT "EACH PLAYER HAS A SYMBOL (* OR #) TO SHOW THE LOCATION OF"
2620 PRINT "HIS LIVE PIECES."
2630 PRINT
2640 PRINT "THE SYMBOLS (* AND #) ARE REGARDED AS THE SAME EXCEPT WHEN"
2650 PRINT "DECIDING ON THE OWNERSHIP OF A NEW PIECE.  AN EMPTY CELL"
2660 PRINT "HAVING TWO # AND ONE * FOR NEIGHBORS WILL GENERATE A #."
2670 PRINT "THAT IS, THE NEW PIECE WILL BELONG TO THE PLAYER WHO HAS"
2680 PRINT "THE MAJORITY OF THE THREE ADJACENT PIECES."
2690 PRINT
2700 PRINT "TO BEGIN, EACH PLAYER POSITIONS THREE LIVE PIECES ON THE"
2710 PRINT "BOARD BY TYPING IN THE COORDINATES (ROW,COLUMN) OF THE"
2720 PRINT "CHOSEN CELLS.  A COMMA MUST SEPARATE THE TWO NUMBERS."
2730 PRINT
2740 PRINT "AFTER EACH GENERATION HAS BEEN PRINTED OUT, EACH PLAYER IN"
2750 PRINT "TURN PLACES ONE PIECE ON THE BOARD.  IF BOTH PLAYERS CHOOSE"
2760 PRINT "THE SAME LOCATION, THAT CELL IS LEFT EMPTY."
2770 PRINT
2780 PRINT "THE OBJECT IS TO SURVIVE.  THE GAME ENDS WHEN ONE OF THE"
2790 PRINT "PLAYERS HAS NO LIVE PIECES."
2800 RETURN
2810
2820 END
READY
```



The commonest stable forms



teaching math with OZ graphics

BY HARVEY COHEN WITH DAVID GREEN

Harvey Cohen and David Green are working with the OZNAKI Project in the Mathematics Department at La Trobe University in Bundoora, Victoria, Australia.

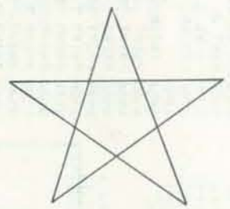
The project, which has the support of the University's Education Research and Development Committee, uses computers as part of a math lab environment for kids. Here the project's graphic language for kids is described.

INTRODUCING OZ-GRAPHICS

'Oz-Graphics' is a computer language which allows the user to sketch an astonishing variety of patterns on a graphic screen using simple commands. It is one of a group of languages and teaching devices, developed by the OZNAKI PROJECT at La Trobe University, which are designed to help teach basic mathematical ideas and problem-solving methods, and to provide an opportunity for mathematical creativity. Oz-Graphics is really a form of computer geometry that is closely related to the 'turtle geometry' developed at M.I.T.

Oz-Graphics directs the happenings on a fine-grained graphics screen (Tektronix GT1024). On the screen is an obedient 'NAKI' (visible as a flashing dot) which can leave a track on the screen as it moves. The NAKI obeys the same moving commands, such as step forward or turn right on the spot, that are familiar to the child and that he could obey himself. A summary of these commands appears on page 57.

By learning to guide the NAKI, the student learns mathematical ideas in a dynamic way. For instance, in drawing regular figures like the following the child learns by experience about angular relationships.



Other mathematical ideas the system demonstrates include operations, group-theoretic ideas, algorithms, and spatial concepts: translation, rotation, reflection, scaling, perspective, and basic geometrical ideas. Oz-Graphics provides a way of doing geometry that helps to develop important analytic skills: detecting repetition in complex patterns, breaking problems down into simple elements, and recognizing and evaluating different approaches to the same problem. It is intended that the system should foster creativity by encouraging the student to design his/her own patterns.

Oz-Graphics is not a C.A.I. system designed to replace human teachers. Rather it is a piece of equipment for a Mathematics Learning Laboratory in which students practice the skills needed to become creative mathematicians. It is very much up to the teacher, through individual or classroom discussion, to reinforce the concepts that children learn on the Oz-Graphics system.

THE WIZARD

When running the Oz-Graphics programme, the user can be thought of as talking to a 'Wizard' who can transmit commands to the NAKI, answer questions, and remember things.

Commands. The wizard conveys commands to the NAKI. Commands are not obeyed until the instruction 'DO IT!' is given. The 'do it!' instruction is given in Oz-Graphics by pressing the RETURN key on the keyboard. This action will be denoted in what follows by the symbol '!'.

Memory. The wizard can be told to remember lists of commands. This is done by typing Z followed by the name of what is to be remembered, then a string of commands. There are four names that can be given to strings of commands you want remembered – V, W, X, and Y. For instance Z V J F 90R F ! tells the wizard that the command V! is to mean J F 90R F !

The command ZZ! tells the wizard to 'remember to remember'. This asks that a record be kept under your name of what V,W,X, and Y represented. They will then be available at the next session you have on the Oz-Graphics system.

Questions. The wizard will take any command preceded by Q as a question. For instance in the example given above, if you type in QV!, the wizard will respond by answering: 'V = J F 90R F.' If the question is meaningless or cannot be answered, then the wizard responds with 'Eh! Eh!'.

The superquestion QQ! tells the wizard to ask you who you are. When you give your name, he will check to see whether anything has been stored under your name. If so, your work from last time can be retrieved and the wizard will reply 'Welcome back . . .' If not, the wizard will welcome you as a new chum by replying 'How do you do . . .' followed by your name.

THE NAKI

The NAKI appears as a dot and can be thought of as a small bug that moves about the screen as you tell the wizard to command it. The NAKI is confined to an area on the screen 1024 units square. If sent over the boundary it jumps to the far side of the screen and continues on its way. By the NAKI's 'state' is meant its position and heading together. The command I! tells the NAKI to indicate its current state.

The NAKI has a home state which is initially in the middle of the screen and pointing up. The command J! tells the NAKI to jump to its home state from wherever it is at the time. The command U! tells the NAKI to revert to pointing upwards when in its home state. There are two ways of changing the home state of the NAKI. One is by walking the NAKI across the screen to the desired location, then using the H command: H! tells the NAKI to establish a new home in its current state. The other involves the use of a cartesian coordinate system with origin in the bottom left-hand corner of the screen. The NAKI's home state is initially in the position (500,380) and pointing up. The command ZH 200 300 45 ! (say) tells the wizard to move the NAKI's home to the position (200, 300) with a heading 45 degrees to the right of 'up'.

The NAKI can move on the surface, where it leaves a trail, or can tunnel. Initially the NAKI is on the surface. The command T! makes it tunnel and it will not surface again until the command S! is given.

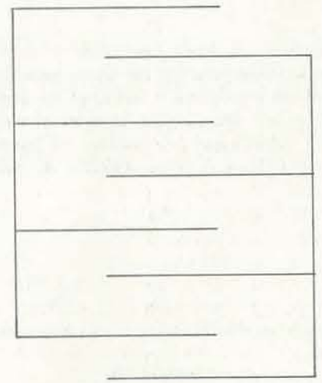
The command F! makes the NAKI move one step forward. A number before the F causes the NAKI to move the appropriate number of steps forward. For instance, 1.53F! sends the NAKI forward 1.53 steps. The step length can be altered to n screen units by typing Z F n !. The initial step length is 100 units. The back command 'B!' works the same way.

The command R! makes the NAKI turn 1 degree to the right of its current heading (i.e. clockwise). L! makes the NAKI turn 1 degree left (i.e. anticlockwise). Larger turns are made by stating the number of degrees first. For example 90R! makes the NAKI turn right through one right angle.

WALKING THE NAKI

Learning about space. Pupils working with Oz-Graphics would normally have previous experience with 'ZONKY', a small robot that crawls about a level surface carring a marking pen. In a first lesson with ZONKY the child learns the basic commands F, B, R, and L by directing ZONKY through a maze. These lessons are worth repeating on-line with NAKI. Commands can be strung together to produce a path showing where the NAKI has walked.

Example: a sample maze



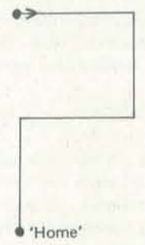
It is impossible to cheat if the NAKI is always on the surface leaving a trail. Of course the NAKI might tunnel through a barrier! The maze itself could be drawn by a slightly more advanced pupil for use by another. Once the NAKI has gone from one end of the maze to the other, a pupil can be asked to make the NAKI retrace its steps back to the start.

Learning about operations. The NAKI has a particular state (position and heading) at any given time. The visible path the NAKI draws is a pictorial record of the state-changing operations used (F B R and L). Thus exercises in which the NAKI must be guided through a maze avoiding the walls sharpen a child's sense of the relationship between operations performed and the state changes observed.

The maze exercises involving retracing the NAKI's steps reveal to the child that F and B, R and L are mutually inverse pairs of operations (i.e. one immediately followed by the other makes the NAKI retrace its last step and return to its previous state). Furthermore, he/she will soon learn that 180R = 180L and that B! is equivalent to 180R F 180R!.

Simple exercises with mazes should make the child realize that replacing each operation in a sequence by its inverse and reversing the order of the operations results in a sequence of operations which causes the NAKI to retrace its steps around the maze.

Example: F 90R F 90L F 90R B ! has the inverse F 90L B 90R B 90L B !.



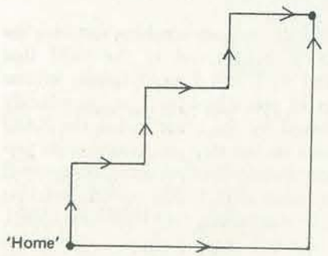
In the example above, note that while the NAKI may retrace its path to return to its original state, it may also return to its original state by an infinite number of other paths.

SPATIAL CONCEPTS

Towards cartesian coordinates. Moving the NAKI about its territory along paths that involve solely vertical or horizontal motions leads naturally to the idea of coordinates that describe its position. Thus if home is taken to be the origin of the coordinate system, then the

NAKI's position can be described as the number of steps ' up ' and to the ' right ' it must make from home to reach that position.

Example: The command strings F 90R F 90L F 90R F 90L F 90R F ! and 90R 3F 90L 3F ! both take the NAKI to a point that is 3 steps above and 3 steps to the right of home.

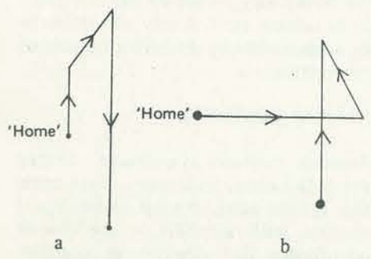


A sense of distance arises from the idea of path length. While the first route in the above example is more tortuous, the number of steps involved is the same along either route.

Angles and distances. Innumerable spatial relationships can be learnt by trial and error on walks with the NAKI. Each step can be tested first by making the NAKI tunnel while performing both the intended operation and its inverse, before trying the operation on the surface.

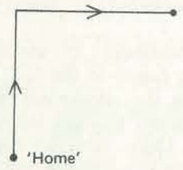
Problem Examples:

1) Give the NAKI the following commands. In each case send the NAKI home without retracing its path. Use only commands contained in the given string.
   a) F 40R F 140R 3F!
   b) 90R 2F 120L F 30R 2B !



The above highlight the geometric properties of angles, parallel lines etc.

2) Give the NAKI these commands: F 90R F !. Try to send it home along a straight line.



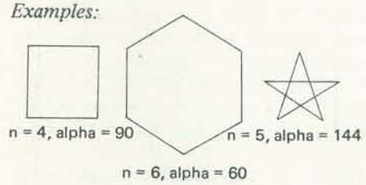
Exercises like (2) illustrate properties of right triangles and trigonometric relationships.

Closed figures. Many geometric ideas can be learnt and many fascinating figures drawn in studying closed figures. The simplest way to draw a closed figure is to repeat the operation of drawing a line segment and turning through an angle the appropriate number of times. For instance, to draw a square we need to tell the NAKI to step forward and change its heading by 90 degrees to the right four times. This is best done by labelling this short string X and giving the command 4X! as follows:

```
Z X F 90R !
J 4X !
```

Theorem. A closed figure will be drawn by a programme like the above provided that the product (n x alpha) of the angle by which the heading changes at each step (alpha) and the number of repetitions (n) of X is a multiple of 360.

Examples:

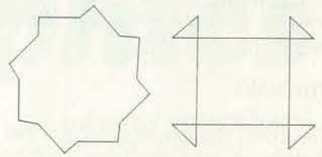


Some fascinating questions arise which students can investigate. For instance:

1) What happens if n times alpha is not a multiple of 360?
2) What happens if n times alpha equals 360?
3) What happens to the figures drawn in question (2) as n becomes larger and larger? (Note: use smaller segment lengths)

Patterns formed by the repetition of more than one line segment pose more of a challenge and give more scope for teaching geometrical ideas. What conditions must now be fulfilled by an algorithm to produce a closed figure?

Examples:



ALGORITHMS AND PROGRAMMES

The essence of pattern is the repetition of basic elements. The V,W,X,Y labels are specifically designed to help in the construction of figures with repeated elements. The programme above for drawing closed figures is an example of their use.

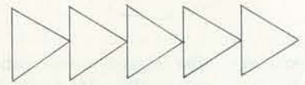
The idea of an algorithm can be introduced by considering command strings with repetitions. For example, in drawing a triangle using the string F 120R F 120R F 120R!, the commands F 120R are simply repeated three times. These two commands can be made into an algorithm named X by typing Z X F 120R !. Thereafter the command 3X! will draw a triangle.

In effect, by having the wizard remember that 3X! draws a triangle, we are teaching the NAKI a new skill based on what it can already do. The V,W,X, and Y memories make it possible to programme the NAKI to draw very complex patterns on a single command. From an educational viewpoint, this feature of Oz-Graphics is probably the most important of all because:

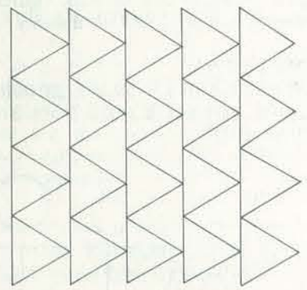
1) It shows the power that algorithms have;
2) It stimulates an analytic approach to problems. That is, it forces the student to look for repeated elements in a pattern and to try to break problems down into simpler subproblems;
3) It emphasizes the use and importance of symbolism in mathematics through the naming of operational strings;
4) It provides a framework within which a student can attack complex problems and, starting from initially rough ideas gradually refine them to reach a solution.

Examples of Programmes:

1) The command X! draws a row of five triangles.
   V = F 120R
   W = 3V T 90R .87F 90L S
   X = 5W



2) If X in (1) is changed to X = 5W J T B S H, then it makes the NAKI return to its starting point after drawing the triangles and move one step down the page. If we make Y = 4X, then the command Y! draws the following wallpaper pattern.



SPATIAL PATTERNS

The following sections outline concepts that are fundamental to the analysis and construction of complex patterns.

Translation. ' Translation ' means a simple displacement of a pattern element in space. For instance, W in example (1) of the last section involves translation of a triangle horizontally. X in example (2) causes vertical translation of a whole row of triangles.

Problem Examples:

1) The programme V = F 90R, W = 4V causes NAKI to draw a square on the c.....and W!. Use this to design a programme to draw a ' chessboard ' (that is, an 8x8 grid of squares).

2) The algorithms V = 8B, W = 90R 8F 8B 90L cause the NAKI to draw vertical and horizontal lines respectively. Use them to design a programme to draw a chessboard.

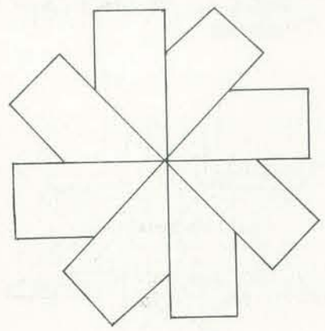
3) Which of the two chessboard-drawing programmes above is faster (i.e. has to do the least drawing)? Which programme is the more compact of the two (i.e. can be expressed more briefly)?

Examples such as the above show that various approaches to problems normally exist and can be used to encourage both flexibility and critical assessment of different approaches.

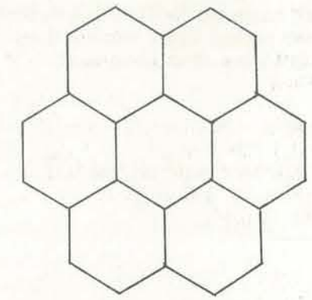
Rotation. ' Rotation ' means repetition of a pattern element by rotation about a pivotal point. This concept is closely linked with the formation of closed figures discussed earlier. The difference is that in rotation the NAKI returns to its starting point each time, while the figures earlier were produced by both rotating and translating the elements (i.e. sides).

Example Exercises:

1) V=2F 90L F 90L F J, W=V 45L H, then 8W!.



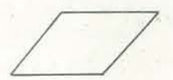
2) Write an algorithm to draw a hexagon. Make use of rotation to write a programme that draws the following section of honeycomb.



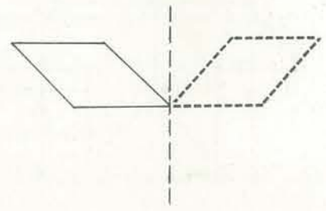
This pattern has sixfold (rotational) symmetry which will find expression in the Oz-Graphics programmes that draw it. There must be some command W such that 6W ! draws the whole figure. Although there are seven hexagons in the figure, if each repetition of W drew one side of the inner hexagon, then there would be no need to draw the inner hexagon separately.

Reflection. ' Reflection ' means the production of a mirror image of a pattern element. Oz-Graphics allows a parallel to be drawn between patterns that are mirror images of each other and the dual nature of the operations F and B, R and L.

For example, the algorithm X = 45R F 45R F 45L B 45R B draws this pattern:

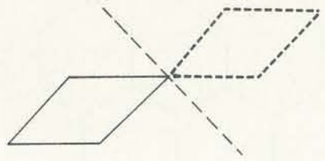


Replacing each R by an L and vice versa give Y = 45L F 45L F 45R B 45L B which draws an image that is the reflection of the original figure about a vertical line through home.

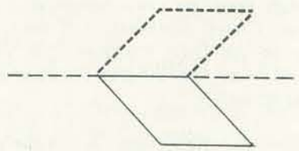


Replacing each F by a B and vice versa gives V = 45R B 45R B 45L F 45R F

which causes the NAKI to draw an image of the original figure reflected about a line at 45 degrees to the vertical.



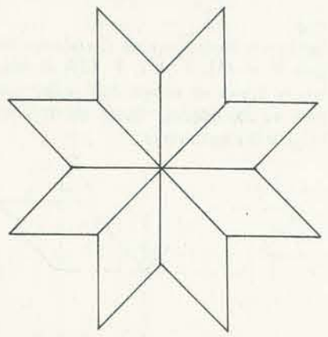
Replacing both F by B and R by L (etc) gives W = 45L B 45L B 45R F 45L F which causes the NAKI to draw an image of the original figure reflected about a horizontal line.



Examples and Problems:

1) Why are the above patterns reflections about the particular lines shown? That is, what is the relation between line of reflection and the operations involved?

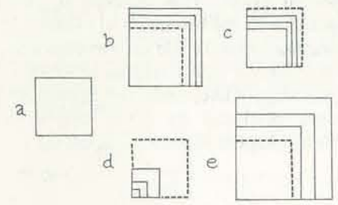
2) Use the algorithms given to illustrate reflection (X and Y will do) to construct a programme that draws the following pattern.



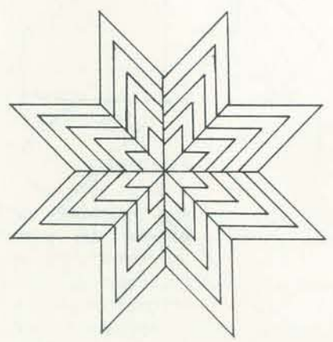
Scaling. ' Scaling ' means repetition of a pattern element with different size. Oz-Graphics has three commands which perform such picture manipulation for any algorithm (X say): G, D, and E. These act like adjectives and modify the command they precede. The form is mGnX (using G and X as examples) where m is a natural number and n is a positive decimal number. This causes algorithm X to be repeated m times and to be scaled each time by a factor n. G causes the figure drawn by X to grow in size by the scaling factor n of its initial size. D causes it to diminish similarly. E causes the figure to exponentially increase in size by the fraction n of its previous size.

Examples:

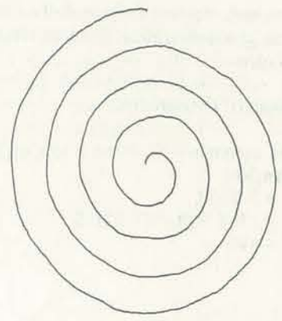
1) If X draws (a), then 3G1.0X draws (b), 3D1.0X draws (c), 3E.5X draws (d), and 3E1.2X draws (e). In examples, the dotted segments indicate the figure drawn by X.



2) Use the algorithm X from the section on reflection to draw the following



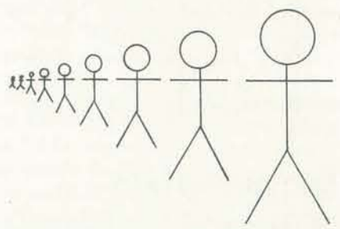
3) Draw an Archimedean spiral



Perspective. The expand command, E, makes it possible to illustrate perspective with Oz-Graphics. Repetitions of figures drawn some distance from home give the same effect as perspective does when a row of such objects is viewed vanishing into the distance.

Example: The programme W! below draws a picture of a stick figure some distance from home. The programme V! draws a whole row of stick figures apparently vanishing into the distance.

```
V = W 9E.7W
W = T 90R 3F S F .5B 90L .2F 90R X J
X = 9L 20Y 81L B 30R B F 60L B
Y = .1F 18L
```



Another aspect of perspective concerns the changing appearance of shapes viewed from different angles in three dimensions. Oz-Graphics has the command M which can be used to demonstrate simulated rotation of a figure in three dimensional space. In the example below, the command 6M15Y! causes the NAKI to repeat the algorithm Y six times. Each repetition is performed as if the NAKI were moving on a surface coming out of the screen and meeting the screen along a north-south line through home. The figure drawn on the screen each time is a projection of the figure on the pseudo-plane back onto the screen.

The pseudo-plane is rotated a further 15 degrees out from the screen on each repetition of the algorithm, so that at the sixth repetition, the pseudo-plane is perpendicular to the screen and the figure drawn is thus merely a straight line along the imaginary intersection of the plane with the screen.

Example: The programme X = .2F 10R, Y = 36X draws a (near) circle. The command 6M15Y! causes a series of ellipses to be drawn.

The M command can be combined directly with the G, D, and E commands above to produce some spectacular effects. It is easy, for instance, to produce patterns resembling views of sea-shells from various angles simply by arranging appropriate rotations and scale changes of an initial figure. The examination of the properties of the shapes of sea-shells is a branch of mathematics in itself and provides a good topic for project work.

CREATING PATTERNS

The skills involved in pattern analysis have already been listed, both in the introduction and in the section ' Algorithms and Programmes '. Exercises to develop analytic ability may take many forms. For instance:

1) Run a given programme to see what it draws. Run each segment individually to see how the pattern is composed.

2) Given an algorithm for an element in a given pattern, try to write a programme that draws the whole pattern.

Creative ability can be stimulated by other sorts of exercise. For instance:

3) Given a pattern and the programme that draws it, see how changes to the programme vary the pattern. Try to produce a given pattern by changing a given programme.

4) Given a pattern element, see what variety of patterns results from programmes that manipulate it in different ways.

Continued on page 58.

SUMMARY OF COMMANDS

CLEARING SCREEN

P	Page! Clears graphics screen Places NAKI at home

NAKI LOCATION

I	Indicates current NAKI heading by a line of flashing dots emanating from the NAKI

NAKI MOVEMENT COMMANDS

F,nF	Forward!, n steps forward!
B,nB	Back!, n steps back
R,aR	Turn right (clockwise)! Turn a degrees right
L,aL	Turn left (anti-clockwise)! Turn a degrees left
J	Jump Home! Leaves no trail on the way home
C	Crawl Home! Leaves a trail if NAKI is "on surface"
H	Redefine home to be NAKI's current state
U	Twists NAKI to head Up screen (North) in home state

TRAIL MARKING BY NAKI

T	Tunnel! Leaves no trail
S	Surface! Return to the surface. Leave a trail

MEMORY

V,W,X,Y	Command strings definable by the user
Z	Remember . . . Is . . . Example: ZX 2F 3R! May also be used to redefine movement commands. Examples: ZH 100 200 90! ZF 50!
Q	Question! Asks the Wizard for information: QX,QY,QV,QW,QH,QF
ZZ	Remember to remember! Store your work for next time
QQ	Superquestion! Tells wizard to ask your name and to retrieve your previous work, if any

PICTURE MANIPULATION COMMANDS

G	Grow at arithmetic rate
D	Diminish at arithmetic rate
E	Expand gives growth at exponentiating rate
nGmX	Do X n times, growing all paths traced out by the factor m of original each time
nDmX	Do X n times, reducing all paths traced out by the factor m of the original each time
nEmX	Do X n times, expanding all paths traced out by the fraction m of current length each time

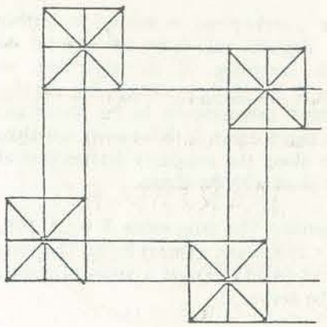
THREE DIMENSIONAL PICTURE MANIPULATION

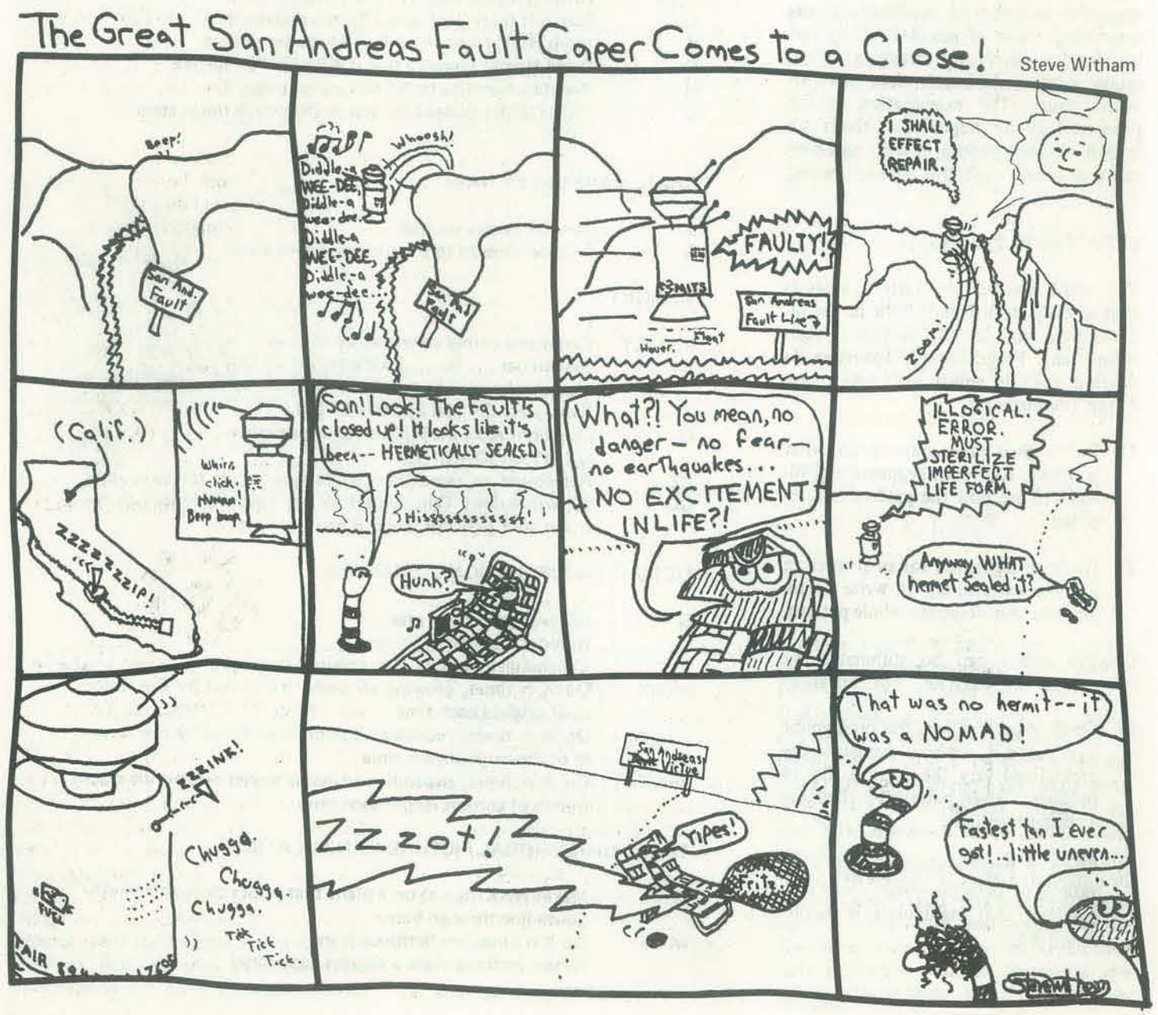
M	Makes NAKI travel on a plane that passes through a North/South line through home
nMaX	Do X n times, on fictitious plane Rotate inclined plane a degrees each time!

Examples of exercises of all the above types are scattered throughout the preceding sections. The nested arrangement of algorithms within programmes that the V, W, X, and Y labels allow encourages good working habits and simplifies the ' debugging ' process when a pattern goes wrong.

Pattern-building lends itself to project work. For example, a project might start with a challenge pattern such as the wallpaper design, right. Difficult patterns like this take time to master. The teacher should encourage students to ask: 'What are the repeated bits of the pattern and how are they strung together? '. Students should be encouraged to solve simplified versions of the challenge pattern first. Students might go on from their final solution to the challenge pattern to investigate other wallpaper designs.

Of course some children will use Oz-Graphics in simple ways to reproduce caricatures of children's art. Thus combining an algorithm X that draws a square with an algorithm Y that draws a triangle, yields a house. The house might be embellished with windows or a chimney. Exercises like this make a pleasant game for younger students. □





ANNOUNCEMENTS

HARDWARE

RS-232 TO CURRENT LOOP/ TTL ADAPTER

The Connecticut MicroComputer ADApter has two circuits. The first converts an RS-232 signal to a 20 ma current loop signal, and the second converts a 20 ma current loop signal to an RS-232 signal. With this device a computer's teletype port can be used to drive an RS-232 terminal, or vice versa, without modification of the port. ADA can also be paralleled to drive a teletype or RS-232 printer while still using the computer's regular terminal. ADA can easily be modified to become an RS-232 to TTL and TTL to RS-232 ADApter. ADA does not alter the baud rate and uses standard power supplies with very low current requirements. The unit comes with complete instructions, is assembled and tested, and measures 3" x 3 1/2" x 1". The current loop is isolated from the RS-232 signal by optoisolators.

ADA sells for $24.50 with drilled, plated-through solder pads for all connections, or for $29.50 with barrier strips and screw terminals. Contact: Connecticut MicroComputer, Pocono Rd., Brookfield, CT 06804.

INTELLIGENT REMOTE CONTROLLER FOR S-100 SYSTEMS

Mountain Hardware's new Introl™ is a remote control system that communicates over the standard 110 VAC power lines. The AC Controller™ board is an S-100 compatible board that is capable of controlling up to 64 remote units anywhere in your building. The AC Remote™ unit has two independently controllable AC sockets that can turn two 500-watt appliances on or off. The computer can also 'poll' the remote to check its status (on or off). Programs can easily be written in BASIC or assembly language to monitor and control remote devices.

The AC Controller™ board in the computer transmits and receives signals to and from the 110 VAC line through an AC Interface Adaptor that is plugged into a wall receptacle. Digital signals generated by the AC Controller board contain an address that selects the proper remote device and a command that turns the remote device on or off. The AC Interface Adaptor 'impresses' this digital signal on top of the 110 VAC waveform and isolates the computer from the AC voltage.

AC Remote™ units, capable of controlling electrical devices and appliances, may be plugged into any wall receptacle in the location. The AC Remote unit decodes the digital signal on the AC line and turns the devices plugged into it on or off in response to the command it receives from the AC Controller board. When 'polled' by the controller, the AC Remote sends a message back to notify the computer of its present state. Each AC Remote unit contains two independently controllable 5 amp AC receptacles and the circuit board, all enclosed in an attractive walnut cabinet. Applications for Mountain Hardware's Introl system include home security, solar heater control and even an easily implemented automated darkroom. Software routines are provided to help create unique control programs. Future compatible remotes include a dual temperature sensor and an 8-input status sensor, which will allow virtually all applications to be realized.

Introl system components are available in kit or assembled form. All AC Remotes are housed in an attractive walnut cabinet. Kit price for the AC Controller is $149.00 and the AC Remote is $99.00. Contact Mountain Hardware, Inc., PO Box 1133, Ben Lomond, CA 95005; (408) 336-2495.

OSI'S HARD DISK

Ohio Scientific has announced a $6000 74 megabyte hard disk for small computers. The C-D74 provides 35 millisecond average access time to any of 74 million bytes of information. The first drive with 12 tracks on a cylinder without reseeking, C-D74 can access any of 220,000 bytes of information in 5 milliseconds.

C-D74 can store all the records of a medium size company for instant access. And the Winchester technology of the C-D74 means that the drive can run 24 hours a day without worry of disk wear.

The 74 megabyte disk also has important applications in both business computing and research in computing itself. The disk makes small computers practical for much larger jobs than formerly thought feasible. With a 10 millisecond single track seek, the drive has a data transfer rate of 7.3 megabits per second, uses a new non-removable sealed chamber drive with a rotary arm positioner.

The drive, cable, interface for an Ohio Scientific Challenger and OS-74 operating system software is $6000 F.O.B., Hiram, OH. Equipment rack not included. Contact: Ohio Scientific, Hiram, OH 44234; (216) 469-7905.

ALF MUSIC SYNTHESIS BOARD

ALF Products has announced a special version of its AD8 Music Synthesis Board. The board has the full features of the AD8 but includes an S-100 compatible 'controller' which contains a top octave generator. Ribbon cables with edge connectors are used to connect this 'controller' to 1 to 8 synthesis boards. Unregulated power is also supplied to the boards which will not plug directly into the S-100 bus.

One example of the board's capabilities is the approximation of a trumpet tone. Twenty-three waveforms defined in the control board's RAM are sent to the synthesis board at a rate of 64 per second. The synthesis board has two waveform memories so that one can be reprogrammed while the other is playing, then on command the board will switch to the other memory as soon as the first element is reached, causing a smooth transition to the new waveform, especially if the zero crossing point of both waveforms is defined as the first element. Although the S-100 'control board' does not have its own processor to send waveform definitions, the computer's 8080 can do this itself. The waveform memories will be accessed as if they were S-100 memory.

The Synthesis Boards are available in kit form for $220 from: ALF Products Inc., 128 S Taft, Lakewood, CO 80228, (303) 234-0871.

PROGRAMMABLE CHARACTER GENERATOR

Objective Design announces the Programmable Character Generator for S-100 computers. This new S-100 card adds the ability to dynamically create the characters generated by a video display device. For those who require special mathematical or scientific symbols, APL characters, sub- and super-scripts, high density bar graphs, greek letters, or game characters such as space ships, the Programmable Character Generator allows the creation and storage of the new characters while retaining intact the original character set. The original character set remains available for use at any time.

Keyboard interface and dual joystick interfaces are provided on the board. The Programmable Character Generator is an ideal addition to SOL (TM) terminals, the PolyMorphic (TM) VTI, the Processor Technology (TM) VDM-1, the Solid State Music (TM) Video Board, and other video display devices utilizing the Motorola (TM) 9 x 7 matrix character generator. Price for the board alone is $44.95 and in kit form $159.95. For additional details, write Objective Design, Inc., PO Box 29325, Tallahassee, FL 32304.

EXPANDOR'S BLACK BOX PRINTER

Expandor, Inc has announced a printer for the computer hobbyist. The Black Box Printer is a low cost ($396.00), fully assembled, 80 column, 10 character per second impact printer. The unit uses a print cylinder (not a dot matrix) containing a 64 ASCII character set and up to three copies are possible on tractor (or pressure) fed 8 1/2" wide paper The printer is shipped ready to connect to (almost) any microprocessor parallel port. It has a parallel interface included, requiring 7 data bits, a ready and a strobe. AC power is supplied for the printer, but the TTL logic interface requires +8 to +10 VDC from the micro.

Full documentation is supplied with the printer including trouble shooting guides, installation and maintenance instructions, printer and interface schematics, plus instructions on how to wire up to the I-O parallel port. Expandor states that most users are able to service their own printer

with the documentation supplied. However, Expandor does in-shop service, or provides parts to the user. A 90 day warranty is offered.

The Black Box Printer is 4.5"H, 13"W, & 10"D, and weighs 11 lbs. The printer may be pulsed character by character with the last character and line visible. The only option is the base and cover for $29.95. Otherwise, the printer is shipped complete – ready to connect and use. Detailed literature is available from Expandor, Inc., 612 Beatty Road, Monroeville, PA 15146; (412) 373-0300.

INTEGRAL IMPACT PRINTER

Integral Data Systems, Inc., is now offering a full-feature dot matrix impact printer designed for use with mini or microcomputer systems. Printing at rates to 120 cps with up to 132 characters per line, the Integral Impact's standard features include an RS-232 and current loop serial interface, enhanced mode (double width) characters, selectable character and line sizes, and multiple copy capability on both fan-fold and roll paper.

The Integral Impact is a complete printer system ready to plug in and operate. Using the RS-232 serial interface, the printer can be integrated into any mini or micro-computer system by simply attaching it to a standard serial port. Serial baud rates of 110 to 1200 bits per second are selectable, and a parallel interface capability is also provided. Switch settings select character sizes and line length from 80 to 132 characters per line.

A 5 x 7 dot matrix is used to print the standard 64 character ASCII set. The print mechanism automatically re-inks the ribbon to give an expected ribbon life of up to 10,000,000 characters. Line buffering by the microprocessor-based controller allows instantaneous print rates of more than 120 cps with sustained throughput of more than 75 cps possible.

Unit price for the Integral Impact is $745 with quantity discounts available. Delivery is 30-60 days ARO. For further information, contact Integral Data Systems, Inc., 5 Bridge Street, Watertown, MA 92172; (617) 926-1011.

VIDEO DISPLAY

The Micro Systems Development MSDV-100 Video Display System is an 80 character, 24 line video output device for the S-100 bus. The character set includes upper and lower case characters as well as full punctuation. Any character can be underlined, and a character can also be made to blink at a user selectable rate, often used for alarm or warning situations. Additionally, a character can be made to appear brighter than normal or to appear in reverse field (black on white).

Also included in the MSDV-100 is the ability to generate high quality forms overlays. Margins can be either single or double width with continuous intersections. Charts, graphs, or order entry forms are easy to produce on the video screen.

A third significant feature of the MSDV-100 Video Display System is the ability to display continuous grey scale elements in any of nine levels in any of 1920 positions on the screen. This is especially useful for bar graphs and for grey scale graphics or animations, as well as in forms applications.

MSD also has the capability to generate and deliver MSDV-100 Video Systems with custom character sets as defined by the user. This could include mathematical symbols, APL characters, or Boolean logic symbols to name a few.

Internally, the MSDV-100 is a two board S-100 based system which occupies 2K of RAM address space and two Input/Output ports. For diagnostic purposes a memory test can be performed on the screen.

Software support for the MSDV-100 is complete with both machine language code, including fully commented source listings, and a comprehensive BASIC software package implementing all MSDV-100 features. The assembly language drivers allow the sophisticated user to easily customize the system for specialized applications.

Programs are provided that permit the user to link the video system to high level programming languages such as BASIC. A link program, provided in BASIC, permits the user with no knowledge of assembly language programming to immediately obtain video output from that software. The link fully implements the forms capability of the MSDV-100, provides direct cursor addressing, and is fully upwards compatible with the LSI ADM-3A video terminal. Price of the kit is $285. Assembled units also available. Contact: Micro Systems Development, 2765 So Colorado Blvd, Suite 110, Denver, CO 80222.

SOFTWARE

WORD PROCESSING

Interactive Data Systems has developed a word processing system, IDSWORD1, designed to run under MITS Disk Extended BASIC. Some of the more important features of the system are:

Line editing – inserting, deleting or changing text in a line of data.

Global editing – inserting, deleting, changing or finding strings of data in a selected block of text.

Merging – combining portions of various files into a single file.

Reformatting – moving words between lines for maximum line size.

Moving text – moving or copying a selected block of lines from one place to another in the text.

Printing – text is printed with optional page numbering and right justification. User specifies left margin, spacing and maximum lines per page. Top and bottom margins are set automatically.

Form letters – multiple copies of a form letter, and mailing labels, may be printed from name and address files.

IDSWORD1 is a package consisting of several programs. This fact is transparent to the user but allows it to run on a computer with 28K of memory. The user selects the mode of operation from a menu list and the control program executes the appropriate program and sets control back to select another mode.

Documentation is extensive and includes many examples and operating hints. The system is provided on a diskette. The total price for the package is $250. Contact Interactive Data Systems, PO Box 290, Owings Mills, MD 21117; (301) 486-6945.

NEW CROMEMCO SOFTWARE

HUH Electronic Music Productions has announced the availability of several new Cromemco software products. The software is supplied on CUTS (Computer User's Tape System) compatible cassettes. CUTS cassettes may be directly loaded into SOL/ZOL or into any other computer with a CUTS cassette interface installed. (Several companies are now manufacturing CUTS compatible boards.)

ROS (Resident Operating System) is a powerful development tool for Z-80 machines. It consists of a complete assembler (Zilog compatible) and Text Editor. It also provides useful system functions such as display, modify, verify, and move memory locations as well as the ability to program 2708 type EPROM's using a Cromemco Bytesaver Board. ROS resides in 8K of memory and is supplied with a comprehensive user's manual, tips for use with ZOL and cassette. Price is $40.00 and delivery is from stock. Dealer discounts available.

Control BASIC is a much extended version of Dr. Li Chen Wang's Palo Alto Tiny BASIC (see *Dr. Dobb's Journal* Vol. 1 #5). Features include: multiple commands per line, extensive output formatting including hard and soft terminal widths, numerical field width, tabs, overprinting, decimal or hex output etc., string input and output with arrays automatically dimensioned, Input and Output commands for direct I/O control, Get and Put for access to any memory location, ability to call user written subroutines with unlimited numbers of arguments.

Also available is a powerful Z-80 monitor. It fits in 1K of memory and allows the user to display, verify, move, and modify memory locations, program EPROM's, display and modify all registers, set up to 5 breakpoints, and many other features. It is supplied on a cassette which contains two separate versions, one for ZOL I/O and the other for other S-100 bus machines. Also included is a comprehensive user's manual which includes the source listing. Price is $25.00 and delivery is from stock.

For more information, dealer program and orders contact: HUH Electronic Music Productions, P.O. Box 259, Fairfax, CA 94930; telephone (415) 457-7598.

MAILING LABEL SYSTEM

Tylog Systems announces the Micro-Label System, a general purpose mailing label system for the creation and maintenance of mailing label lists and other similar label applications (such as inventory bin labels, etc.). The system is designed to be run by non-technical personnel. Extensive documentation and procedures are provided to facilitate operation of the system.

The software is designed to run on an 8080 processor using dual North Star Computers' minifloppy disk drives and a pin feed or tractor feed impact printer. Additions, changes and deletions are applied to the master file via CRT. The Micro-Label System permits the label files to be printed in more than one sequence. Error editing and restart capabilities are also provided. Each master file can hold over 540 records with multiple files being used for longer lists.

Tylog warranties the software for 90 days. The warranty includes new release updates at no extra charge. Maintenance contracts are available for post warranty support. Price is $500 for the standard package, customized options are $40 each. The Micro-Label System is also available as a turnkey package, complete with all hardware.

For details and your nearest Tylog dealer write: Tylog Systems, Inc, 9805 SW 152 Terrace, Miami, FL 33157.

MICRO APL CONTEST

BYTE Publications, Inc., has announced 'The Great APL Interpreter Contest.' One or more $1000 prizes will be awarded to authors of contest entries, which must be postmarked no later than midnight, February 28, 1978. Contestants are free to write their interpreters for any microprocessor they choose. Entries will, however, be judged on their suitability for use on small systems with a minimum of 16K bytes of memory, as well as on programming elegance and efficient use of space. Entries must be in the form of a publication quality manuscript which describes the implementation of the interpreter and which includes source code and object code. Contestants should also submit machine readable source and object code in the form of paper tape or cassette tape. Judging will be done by the editors of *BYTE* magazine. Those seriously interested in entering this contest should call Carl Helmers, or Chris Morgan at BYTE, (603) 924-7217 in order to be included in mailings of further information about the contest's progress. Or write BYTE Publications Inc., 70 Main Street, Peterborough, NH 03458.

ANSI APPROVES MUMPS STANDARD

The American National Standards Institute (ANSI) has approved the specification of the MUMPS Language as an American National Standard. MUMPS thus becomes the third computer language, after FORTRAN and COBOL, to be so approved. MUMPS (the Massachusetts General Hospital Utility MultiProgramming System) is a general-purpose, interpreted programming language designed for interactive data management applications. It features a comprehensive set of operators and functions for manipulation of variable-length string data, and a symbolically referenced, hierarchically structured, shared database.

Development of MUMPS began in 1966 at the Laboratory of Computer Science, Massachusetts General Hospital, Boston, Massachusetts. By 1972 no less than six dialects were in common use for educational, medical, and commercial applications. At that time, the National Center for Health Services Research of HEW and the National Bureau of Standards jointly sponsored two major efforts: the MUMPS Development Committee, charged with defining a MUMPS Standard that could be accepted by current users of the individual dialects; and the MUMPS Users' Group whose responsibility was to promote the use and availability of the MUMPS language. The MUMPS Development Committee, which now serves as sponsor and development body for MUMPS, completed the Standard specification in September, 1975. This specification was submitted to ANSI for approval via the canvass method. Under this procedure, all groups having a substantive interest in the proposed standard were polled to obtain a consensus on the quality and appropriateness of the specification. Voting results were subsequently submitted to the ANSI Board of Standards Review with final approval occurring on September 15, 1977.

MUMPS has enjoyed an annual growth of about 80% in numbers of new installations. In mid 1977 there were 600 or more institutions around the world at which MUMPS was used. While a large percentage of the user base remains medically oriented, MUMPS is finding increased use in business and commercial applications where its flexible interactive data handling philosophy can be effectively utilized. MUMPS is currently available on six major computer lines.

Further information on MUMPS is available from Mr Richard Zapolin, MUMPS Users' Group Executive Director, MITRE Corporation, PO Box 208, Bedford, MA 10730.

GATHERINGS

SAN FRANCISCO, CA, FEB 28 - MAR 3

The IEEE Computer Society presents its 16th international conference, COMPCON 78 SPRING, at San Francisco's Jack Tar Hotel on Feb. 28 - Mar 3, 1978. In addition to the traditional presentation of papers, a special series of evening sessions will be aimed at computer novices who are non-IEEE members. Contact COMPCON, Box 630, Silver Spring, MD 20901; (301) 639-7007.

SAN JOSE, CA, MAR 3-5

The 2nd West Coast Computer Faire will be held at the San Jose Convention Center, San Jose, CA March 3-5. Details are available from Faire Chairbeing Jim Warren, (415) 851-7664, or write The Computer Faire, Box 1579, Palo Alto, CA 94302.

COLLEGE STATION, TX, APR 21-23

On April 21-23, Texas A & M will hold its 2nd Annual Microcomputer Exposition at Earl J. Rudder Tower in College Station. For more information contact A & M Microcomputer Club, Box M9, Aggieland, TX 77844; (713) 822-7118.

LONG BEACH, CA, APR 28-30

The Long Beach Convention Center is the site of PERCOMP '78. For details contact Royal Exposition Management, 1833 E. 17th St., Suite 108, Santa Ana, CA 92701; (714) 973-0880. Or call Dede Ginter who's handling public relations, at (714) 879-9920 (collect calls accepted) for advertising and media information.

OTHER

ASSOCIATION OF COMPUTER RETAILERS

In June, 30-40 computer store owners decided to form an association of computer store owners; Dr Portia Isaacson will serve as chairperson until a national association is formed.

There seemed to be two schools of thought as to the primary purpose of a national association. One was that the association's primary purpose should be to provide services that individual store owners could not provide themselves. Examples of such services include compiling industry ratios and statistics, circulating a newsletter, arranging group health, life, casualty, liability and workmen's compensation insurance, lobbying, providing centralized legal resources and, perhaps most importantly, providing clout with manufacturers. The other school of thought was that the primary purpose of the association would be to develop high standards among computer retailers. Dr Isaacson notes that these two schools of thought are not mutually exclusive and are actually quite compatible.

At the June meeting a proposal was presented by Attorney Kenneth Widelitz to create a financial basis for the establishment of a national computer retailers association. By mid-November, Mr Widelitz expected to have determined whether there was adequate interest to formally establish a trade association. For details on current status of the proposed organization, contact Kenneth S. Widelitz, 10960 Wilshire Blvd, Suite 1504, Los Angeles, CA 90024; (213) 477-3067.

TRS-80 USERS' GROUP

The TRS-80 Users' Group is dedicated to the exchange of programs and technical data for the new Z-80 Based System by Radio Shack. Those wishing to join the group may send a self-addressed stamped envelope to: R. Gordon Lloyd, 7554 Southgate Rd., Fayetteville, NC 28304.

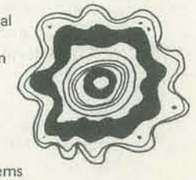
CENTRAL STANDARDS LIBRARY

To help solve some of the standards problems in the hobbyist computer and microcomputer field, ALF Products is sponsoring a Central Standards Library. After discussions with several manufacturers in this field at the West Coast Computer Faire, ALF has set up the CSL as a means of standards information exchange for manufacturers, consumers, hobbyists, and others interested in standards. The Library will collect submitted standards and distribute them on a non-profit basis. For more information on available standards, on how to submit standards, and on the Library's services; send $1 (to cover printing and mailing costs) to: The Central Standards Library; c/o ALF Products Inc.; 128 S. Taft; Denver CO 80228. You will receive a copy of the first CSL Newsletter and the first submitted standard (a parallel interface standard). Manufacturers currently participating include: ALF Products, IMSAI Manufacturing, PolyMorphic Systems, Proko Electronics, Vector Graphic, and Video Terminal Technology.

PERIODICAL GUIDE FOR COMPUTERISTS

The January - June *Periodical Guide for Computerists* indexes 1080 articles from 23 hobby and professional computer publications. Articles, editorials, book reviews, and letters from readers which have relevance to the personal computing field are indexed by subject under 90 categories. The 32 page book is available postpaid for only $3.00 from E. Berg Publications, 1360 SW 199th Ct., Aloha, OR 97005 or from local computer stores. A January - December 1976 Guide is also available for $3.00 postpaid. List of magazines indexed:

Byte
Calculators/Computers
Computer Music Journal
Computer Notes
Creative Computing
Digital Design
Dr. Dobbs Journal
EDN
Electronic Design
Electronics
Ham Radio
IEEE Computer
Interface Age
Kilobaud
Mini-Micro Systems
People's Computers
Personal Computing/Microtrek
Popular Computing
Popular Electronics
Radio Electronics
SCCS Interface
73 Amateur Radio



COMPUCOLOR USERS' GROUP

A group of us have formed a Compucolor Users' Group dedicated to the exchange of programs and technical data for the Compucolor. We anticipate issuing a news bulletin periodically. Subjects such as how to concatenate tapes and disks will be covered. For each accepted program a member will receive in return a number of other programs. Our present programs include a right rectification program and illustrated versions of blackjack, startrek and slot machines. We will try to exchange recorded media rather than program listings. An initial membership fee of $10.00 covers cost of duplicating and mailing materials. For further information send a large SASE to Stan Pro, S.P. Electronics, 5250 Van Nuys Blvd., Van Nuys, CA 91401. □