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

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

Byron Johnson
356 Laguna Terrance
Simi Valley, CA 93065

BACK ISSUES!

$1 EACH

GET 'EM WHILE THEY LAST!

Games, listings, programming languages, random insanity! We still have available a few issues stretching way back to the beginning of time (October, 1972) when *People's Computers* was published as a newspaper called *People's Computer Company*.

Highlights of available back issues are listed on the order card at the center of the magazine. Buy a couple at a buck each, or go whole hog and get ALL 15 for just $12!

people's computers

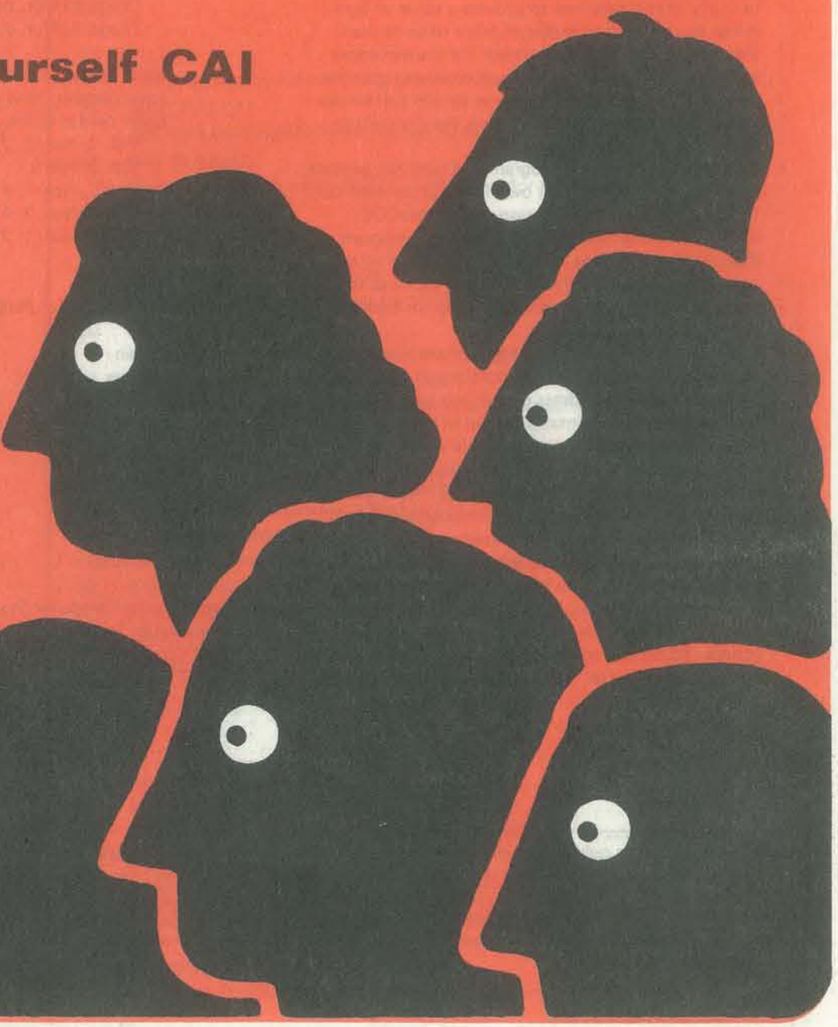
$1.50

VOL 6 NO 1 JULY – AUGUST 1977

HEATHKITS 8080 & LSI-11

Do-It-Yourself CAI

ROBOT PETS



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.

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, Nos. 1, 2, 3
- Vol. 2, No. 5
- Vol. 3, Nos, 1, 4
- Vol. 4, Nos. 3, 4, 5, 6
- Vol. 5, Nos. 1, 2, 3, 4, 5, 6

Foreign Distributors of *People's Computers*

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

Comicro AG
Badenstrasse 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

Kouqakusha Publ. Co., 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

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 1
JULY - AUG 1977

CONTENTS

LETTERS

Dennis Allison did it again! I think his Tiny PILOT proposal is grand. *(See our Jan/Feb 77 issue, or Dr. Dobb's Journal for March)* I have three little comments:

1) For minimal resources implementation, the DEL, SEE, and CONT commands can be deleted. DEL can be accomplished by typing the statement number followed by a CR as in Tiny BASIC. SEE and CONT can be done by T: and J: respectively.

2) It seems unnecessary to have two kinds of strings.
   a) If only closed strings are allowed, we can use the same convention as in Tiny BASIC. Example:
   "NOW IS TIME TO PRINT $A WHICH CONTAINS", $A
   b) If only open strings are allowed, we can use the @ sign to indicate substitution. Example:
   NOW IS TIME TO PRINT $A WHICH CONTAINS @$A

3) If (2a) or (2b) is adopted, we can further drop the # sign for numerical variables. Example:
   100 A: X
   200 C: Y=X*2
   300 T: "X=",X,"Y=",Y (2a)
   (300 T: X=@X AND Y=@Y) (2b)

Lichen Wang
150 Tennyson Ave.
Palo Alto, CA 94301

Dennis Allison, as a user of your previous works, via Tom Pittman's 6800 version of Tiny Basic, I am very interested in the proposed Tiny Pilot. First reaction – HORAY. Second reaction, same as the first.

I'll have to read it closely to see what it won't do that Tiny BASIC will. If it will do all, more HORAY. I manipulate strings much more than I do numbers, on all computers that I use.

1) Please consider that the only non-alpha character that is under the resting fingers is the SEMICOLON. Consider also that the COLON has meandered around on different versions of Ascii terminals and typewriters. I have COLON 3 different places on 3 machines in my office. Therefore; Please consider semicolon instead of colon for the most-used delimiter in your language.

2) Please put back an external call (peek and poke, or USR). A majority of users of tiny are as higher-level drivers for things being done partly in machine language. I tend to write things COMPLETELY in Tiny, then bit by bit move parts to machine language, called by Tiny, than finally commit to EPROMS, and still use TINY as a caller.

3) I hope it's just typos that left out (and) in the March *DDJ*.

4) If you can't see using semicolon instead of colon, how about using it instead of #, the second most used symbol.

5) It's cute, but very limiting, to have the MATCH statement simply report presence or absence of a match.
100 MY: dog, puppy,kitty,horse for example, still probably makes me wonder whether it's a dog or a horse, with another match statement or two.
The H/P 2000 version is marvelously flexible:
100 A = POS ("dog puppy kitty horse", A$(1,LEN(A$–1)))
says A = the position in the match string of A$ with the last character stripped off (in case he said "dogs"), with A=O if there is no match.

6) PLEASE MAKE IT EPROMABLE. I've reached the point where NO routine which can't be made ROM isn't worth the trouble. And in case anyone hasn't figured out how to address a nearby table in 6800 code, where the P register is not addressable, I submit the following code, which puts the P register in the X. Let those who need it figure it out.
8D 00 30 31 31 EE 00

N.J. Thompson
Hawaii Institute of Geophysics
2525 Correa Road
Honolulu, HI 96822

Dennis Allison replies:

Thanks for your note. Regarding your comments:

1) The choice of colon was historical; it's used that way in the PILOT-73 language and other versions. If it's a problem for you, you could put it on a toggle and let the user specify it. That could, of course, get him into trouble.

2) External calls are important, but are something of a bandaid extension to the language. In my first cut, I left them out. Incidentally, there is a revised specification for Tiny BASIC, unpublished and unimplemented, which supports strings and external functions. That will appear in DDJ sometime this summer. Also, Bob Albrecht and I are working on a new language.

3) A lot of typo's did creep into the published text. My fault too! I did proof it, but somehow in the rush to get DDJ out the changes did not get made. The reason that you got DDJ so late was the Computer Faire; Jim Warren, Faire organizer and DDJ editor, didn't get it together on schedule.

4) Take a look at PCC's version of the Tiny PILOT specification. It's a bit more detailed and has a few minor differences. (It was published earlier – January – than the DDJ version, but was prepared later.) The major difference is that '#' need appear only when a number is wanted inside a string; the usual tiny BASIC convention works fine in assignment statements (variables need not be prefaced by '#').

5) A more complex match statement with an implicit assignment would be nice; we ruled it out because this is *Tiny Pilot*. Some Pilot systems do provide this facility by placing the matched text into a special variable.

6) I agree that all code should be pure procedure (the technical term for code which does not modify itself). It shouldn't be much of a problem.

Dennis Allison
PCC
Box E
Menlo Park, CA 94025

Here are some rules for MacOglesby's game EXAGON, published in the last issue of *People's Computers*.

The board is an order 5 hexagon having 26 randomly selected points labeled with the letters of the alphabet. Every point of the hexagon can be considered to have three lines passing through it, one horizontal and two diagonal, at 60° and 120° from the horizontal. The 2 diagonals form an 'x', hence, you see, the name 'EXAGON'.

A move consists of selecting one of the points labeled with a letter of the alphabet. All the labeled points in the 'x' generated by the diagonals through the selected point (including the selected point) are then relabeled with the mark of the player who moved.

NB (1) you can only select a point labeled with a letter (not with a player's mark)
(2) all labeled points in the x are changed (whether labeled with a letter of a player's mark)

Play continues in turn until no playable (ie letter labeled) points are left. The player with the greatest number of marked points is the winner.

Eryk Vershen
People's Computers
Box E
Menlo Park, CA 94025

Dear Dragon,

Hey mythical monster, keep up the good work on SAM (the make-believe computer). Dragonese is the best teaching language I've come across, and at this time we need to be educated in the workings of the computer more than we need better computers for the selected few to use.

OK, your threat to join Puff the Magic Dragon in some far away cave has caused my footprints to lead to your door. Problems 21 and 22 not only can be solved without SAM's CHS and XCH functions, but also 21 can be solved using fewer instructions.

Problem 21 $a^2 - a$

INSTRUCTION	SR0	SR1	SR2
CLS	0	0	0
INP a	a	0	0
UP	a	a	0
X	a^2	0	0
INP a	a	a^2	0
–	$a^2 - a$	0	0

Problem 22 $a^3 - a^2$

INSTRUCTION	SR0	SR1	SR2
CLS	0	0	0
INP a	a	0	0
UP	a	a	0
UP	a	a	a
X	a^2	a	a
INP a	a	a^2	a
–	$a^2 - a$	a	a
X	$a^3 - a^2$	a	a

J.E. Jobaris
724 Oregon Ave.
San Mateo, CA 94402

Some suggestions on how to improve PCC magazine: Get rid of the news about how schools are using computers and publish more games programs, and more information on new and little-known but useful programming languages. I especially liked the listing of Z-80 tiny PILOT. Let's see more of the same. On the other hand, I don't particularly care for "Flying Buffalo" type games; so I feel that the Don Quixote Starship will probably turn out to be a bummer. Also, your "Make-believe Computers" series is one of the most assinine things that I have ever seen in a computer magazine; I think the whole thing is a complete waste of space. On the other hand, the Dragon's completely self-contained Tiny BASIC computer is just what we need to bring personal computing out of the advanced experimenter's basement and into the family living room (But how about an extended-BASIC ROM and a PILOT ROM too? Tiny BASIC may be OK for kids just learning to program but it is a bit too restrictive for any serious programming.) And we need some tapes with useful programs like checkbook balancers and menu planners too!

And who is going to manufacture this wonderful machine? (Or are you hoping that your articles will inspire some manufacturer to do so?)

Keep up the good work!

Sam Hills
3514 Louisiana Ave. Pkwy.
New Orleans, LA 70125

One of the guys at work made up the enclosed drawing. Thought you'd like a copy. He never even heard of PCC (until I talked to him!)

Keep up the good work!!!

Glen Charnock WB6JKM
864 Palomar Way
Oxnard, CA 93030

I just got the May-June "People's Computers". THIS IS NOT CRAZINESS! Where are the Doomsayers, the letters from little kids, the funny hats, the uplifting prophecies, the "Ng"? The "Women and Computers" was neet partly because the author had a Poly (yey!) with their 11K BASIC, which *is* good software. I have much more to say on the real topic of the letter, but . . . I had already read "Garden of Illusions" from *Earth/Space News,* and I thought it was the best article they ever printed. I still don't understand just what DQS is supposed to be, though. I also noticed the photo of the new Poly. Those guys, too, seem to be getting more serious, less fun.

Anyhoo, my main point is: thanx for printing my description of TONEGEN, the four voice 8080 tone generator. "Media" got printed as "Mecia," but the zip code got four orders through so far.

Let me update you on TONEGEN. The documentation is ten pages (including listings). The demo music is "Thus Spake Zarathustra" (the "2001" theme). The price is now $3.00 + 25¢ for my unfinished (2 "verses") version of the "G minor Fugue". I don't send it out until I've gotten the signed agreement (included at the end of this letter.) I include schematics for mono, stereo and quad output, assuming you already have a parallel outport and amplifier(s). One of my customers may write a smart driver program for TONEGEN – he already has one for "Alphanumeric Music". If so, we would sell it with an arrangement (& low price!) similar to the one with TONEGEN.

Is this "agreement" business puzzling or put-offing? How come I don't just put copyright on my software? Howzabout I explain my philosophy on software distribution, etc.? I do not believe that "twiddle-de-de, ideas should be free", to paraphrase Ted Nelson's paraphrase. But I also don't believe that a person has a right to a monopoly over any market, unless he has gotten it thru *voluntary* dealings with other people. If I take or even use someone's copy of a piece of software without his permission, then, yes, that is stealing. But duplicating a copy that he has *given* me in no way diminishes his control over *his* copy of the software. I don't feel it's *immoral* to sell copies of copyrighted software, only *risky*. In fact, I think it's disgusting to use (threats of) government force, via the copyright *laws*, to restrict other people's actions. It would also be stupid to ask them to respect a "right" which I don't personally believe I have. So what course is left to a libertarian-anarchist who wants to profit from his creative efforts?

The answer, my friends, is in that word, "voluntary". You don't have to use force (law) or intimidation ("respect my rights, you greedy thieving fiend!") to make a profit. After all, customers already voluntarily give up their money for software (do we need a law to make them?), so why not have them agree to let you control the distribution of the software which they buy? In other words, the program is *theirs* – they bought it – but they've *agreed* not to let it go.

Not only is this scheme more honest than using a copyright, it's also more flexible. You can't rewrite the copyright laws like you can a contract. Do I think everybody will stick to their agreements? (why am I asking myself?) No, but I think most people will (I would) – certainly more people than respect the ol' (C). In fact, probably many people *resent* the impersonality of "Copyright 1977 by . . ." Maybe I should put it this way – I want to make as much profit as I can from my programs, but I depend on People (heh, heh), not government, to make it happen. Besides, if I find proof that someone has gone against his agreement in a big way, I'll have his signature to print, along with, perhaps, a letter from someone he's given the software to . . .

Finally, let's get back to the style of *People's Computers*. Look – I subscribe to your magazine because you're DIFFERENT! And now you're turnin' into something NORMAL! Please, I'll do anything! Even contribute!

First, start phasing out those old fashioned woodcuts. Most of them are boring and stereotyped. (Altho some are OK, I like that little wierd lion-animal saying "Ng" on p 19 PCC V5 #4). Also, stop printing "press releases" verbatim from companies. Those things already appear in all the other computermags. I want reviews and comments, not echoes! Here's a guideline: If it's not worth putting in your own words, or you wouldn't want to be held personally responsible for what it claims, then DON'T PRINT IT!

Now for some real contributions – I have a friend who produces all sorts of wierd drawings and comix – are you interested in lots of spacecraft? I'm enclosing "Software Sam", "Rah, Rah, Thrud", and "Capt. Gen'rul-issy-mo-Sarge". I've also included a !Giant Sand-Warthog! of mine. So, hasadassa. That's about it.

Steve Witham
168 Painter Road
Media, PA 19063

Agreement form

I, , agree not to let anyone have a copy of TONEGEN or its documentation unless (1) he makes this same agreement in writing to Steve Witham, (2) either he or I send $3 to Steve Witham as "royalty" for the copy of TONEGEN he is receiving, and (3) I have received from Steve Witham acceptance of the payment and agreement. I also agree not to copyright any material related to TONEGEN.

Signed,

Gee whiz, Steve, only 2 articles besides Reviews and Announcements used woodcuts. And we underpay our artists as is, so we can't depend on all original art. We encourage people to submit original photos or art in black ink on white paper when submitting articles. Other art is also welcome but we don't always find the appropriate place for it. One area in which it would be very nice to have a backlog is cartoons – any good computer-related one-liners you'd like to illustrate?

Readers have indicated that even unedited product announcements are useful to them so we continue to print a few of them in abbreviated form. Product reviews and comments are always welcome, but we don't have the in-house staff to produce them. Contributions from readers are joyfully received.

For new readers, Steve's four-voice polyphonic tone generator subroutine for the 8080 is described on page 45 of our March/April issue, Volume 5 Number 5.

As you may recall, the March-April issue of *PC* ran a short item in which I suggested that a computer might generate stories wherein chess games are converted into accounts of medieval battles. Well it seems that this has already been done, at least in fiction. Rae Montor of Oakland informed me of a story by Poul Anderson in which chess battles are waged by computer-controlled robots. This story, 'The Immortal Game', appeared in the February '54 issue of *The Magazine of Fantasy and Science Fiction* (Vol 6, No 2, p 155-124).

Jim Day
17042 Gunther Street
Granada Hills, CA 91344

Just bought an armful of hobbyist journals in one of these chaotic new computer stores (like Hoboken Computer Works), noticed a dart by Sol Libes somewhere on 'money-making schemes such as a glossy magazine', and was most impressed with the human component and California breeze arising from your candid pulp for a plausible price.

The reason for writing you is a need (as so many readers seem to ask you for help) that suddenly crossed my mind as possibly relatable to the thoughtful kiddie amusements in your Jan-Feb 77 issue, and CAI and computer graphics that are mentioned so much. Believe me, I'm not a reactionary and besides not a connaisseur of comic strips, I respect your reader's expressed opinions, but could we have just a 'Tiny Sprocketman', not on paper, rather on the screen? Perhaps with software created in a competition endowed by the corporations whose products are crowding the roads? (Prizes rather than grants or wages can keep it a hobbyist activity, and provide richer alternatives.)

You may suspect a layman out there expects a microchip to turn into a cinematograph, but I have more diagrammatic action in mind, just some improvement on the static and unresponsive charts of intersections one may see in driving schools or manuals.

The incident which pointed out this need to me was that my son bicycled into a car, wrecked his sister's bike, and was found ignorant of the rules of the road, although a good number of hours of theoretical and (somewhat precarious) practical training in actual traffic had been administered (and he is bright enough to program a pocket calculator). The local police department gives well-informed general safety talks in schools, and registers bicycles (without endeavering to examine or certify their mechanical safety), but has no facilities to train and test youngsters individually.

Is it safe to assume that the people who write and read PC could do quite a bit about this, short of putting a steering wheel on a microcomputer? (Or has a video game manufacturer done it already?)

Good luck with your de-dragoning, or whatever may uphold your social relevance!

Roger Belling
398 Ogden Ave.
Jersey City NJ 07307

No one around here is aware of any CAI bicycle safety course, but it sounds like a great idea. We'd be happy to support and publicize such a contest but it needs a coordinator to set it up. Any volunteers?

I am writing you in the hope that you will be able to help our search for elementary school level computer teaching materials. We are a spiritual community with an elementary school, a computer, and the wish to somehow put the two together. We have professional teachers and programmers, but neither has worked in the other's areas before. Any form of teaching materials, software, etc. would be helpful.

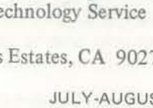
Our computer is a PDP 11/03 with FORTRAN and multi-user BASIC, with floppy disks and a DECWRITER terminal. Our school has about 30 students ranging from age 2 to age 10, and serves both our community and outside children. We are non-profit and have tax exempt status.

We would be happy to share our experience and any materials that we develop here. Thank you for your assistance.

Vakil Kuner
The Abode of the Message
Box 376 Shaker Road
New Lebanon, NY 12525

A most practical problem: How do you convert programs written for a particular microcomputer with a given microprocessor to another microcomputer with the same microprocessor? This is one of the most immediate practical problems faced by a dragon.

What's needed:
- a list of the address space layout for each machine.
- a list of the prom monitor routines – or at least a description of each subroutine (such as imput a hex character), and its entrance and exit addresses.
- a list of each machine's other ideosyncracies.
- instructions on how to use a dissembler to locate all I/O, monitor requests, and other position dependent code.
- general principles for making the modifications.

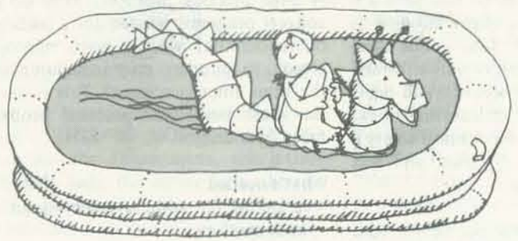


If you wish, I'll send you the above data on the OSI-400 system (I'm a dealer). This is the sort of problem that Tom Pittman had to face – perhaps he has some advice on what to do if one is faced with a program for another machine (with the same microprocessor) where his nine suggestions haven't been followed. (PCC, V5, no. 5, pg. 13 and DDJCCO V2 No2 pg.4). I bought his TB and hope to buy more from him. He will be writing 6502 FORTRAN!

How about MUMPS (see Decus) for a micro? It is a "plain English" interactive language with the ease of use of PILOT and a great deal more computational and data base manipulating power. For example, there are patient-computer interacrive medical diagnosis programs in MUMPS in the public domain which are *better* than the average doctor (this is according to a panel of MD's who are diagnosticians).

If you publish a FORTRAN MAN comic, I'll buy a few! Thanks for your fine product.

Durk J. Pearson, Entrepreneurial Dragon
Spectrum Technology Service
Box 942
Palos Verdes Estates, CA 90274



PET ROBOTS:
New Capabilities

BY ROBERT ROSSUM

We're pleased to present a second article from Robert Rossum of the United States Robotics Society, P.O. Box 26484, Albuquerque, NM 87125. The first article, 'Robots as Household Pets' (also copyright 1977 by the U.S. Robotics Society) appeared in our January-February issue, Volume 5 Number 4.

While it is comparatively easy to build machinery that seems exotic and exciting, it is surprisingly difficult to build really new machines, with new functions, that seem comfortably familiar and leave observers unexcited. Nature has taught human beings over the millenia that new, little understood things are hazardous. Almost all mutations are lethal. Living things are more often harmed by dramatic change than improved by it and the sensible human being properly mistrusts innovation.

Yankee do-gooders have found it difficult to change the ways of poverty-stricken subsistence farmers in distant lands where modern farming techniques could markedly improve the quality of life. While visiting agricultural experts know that their new ideas will work, the farmers know only that they have survived thus far by traditional means and that if they lose a single crop in a single year, their families may starve to death. Under the circumstances, they tend to a conservative view of innovation.

None of us is so far removed from the uncertain life of huntsman and subsistence farmer that he is not frightened by genuinely new things. Since robots will be genuinely new in the human experience, their developers may choose to package robots in comfortably familiar form that cheers and soothes the observer. In *Robots as Household Pets,* we treated the notion that robots might well be modeled after traditional household pets. Floor-loading characteristics, density, weight, smell, taste, colorations, speed of movement, and demeanor might all be determined for robots by close study of animals that are already welcome in the home. (For example, pet robots might have large eyes with large pupils, set beneath large foreheads. That's characteristic of baby mammals, by whom hardly anybody feels threatened. Further, when mammals are upset enough to bite, their pupils tend to narrow to pinpoints. The large pupils of the robot pet suggest a continuous state of non-aggressive calm. We all respond to these little physical cues at a subconscious level. The robot designer who values his property may wish to avoid frightening his neighbors with hard-eyed, unsympathetic adult robots. An inadvertant fright is as upsetting as a deliberate scare.) Much is known about lovable household pets and the knowledge can be applied to robotics in a straightforward manner.

The robot pet could bumble about good-naturedly as the family dog does, though perhaps with less shedding of hair on the carpet and less slobbering on the linoleum in the kitchen. Robot pet will pay close attention to family activities, warn of intruders, and develop fondly appreciated character traits and crotchets.

Apart from extraordinary endurance, sensitivity, and cleanliness, the robot pet might be capable of uncommonly useful performance. It might, for example, be an 'idiot savant'. Some people in the world have unaccountable talents, like the ability to multiply two twenty-digit numbers in their heads with perfect accuracy in seconds. Such people may be able to tell you promptly the day of the week on which Christmas will occur in the year 2315 without any previous preparation. One chap can transcribe on paper all the instrumental parts of music he hears played by a symphony orchestra, just as fast as he can write. He can sing every scrap of

Wagnerian opera, music and lyrics, and recite the dialogue. Though he's in his forties, his social behavior is that of an immature four-year old who whines and weeps if his mother doesn't break off conversation with others to pay attention to him. Many people with such odd talents are in other respects mentally inadequate, unable to care for themselves, unable to make sense of the world around them. The cases are numerous enough so that the term 'idiot savant' has grown to be a useful descriptor.

Idiot savants may be storehouses of vast knowledge that can be elicited from them at will, but typically they are not able to relate the knowledge to anything else, to make appropriate judgements based upon what they know. The robot pet might be able to provide the idiot savant's useful knowledge and skill without the unhappy side effects and the emotional wear and tear. Automatic speech recognition techniques are improving rapidly and the next few years should be filled with demonstrations of remarkable capability in the field. Speech synthesis, similarly, is becoming more common, flexible, and satisfactory. A robot pet may within the next decade be equipped with great powers of speech and speech comprehension.

The robot pet is sure to be the pal of the household children, a faithful and loving companion. Cannot the robot also be a powerful educator and tutor?

'Tell me a story, a good one,' says cheerful child.
'What would you like?' says robot pet, with pleasure.
'I like the one about Saladin's meeting with Richard the Lion-Hearted.'
'Richard and Saladin probably never really met.'
'I know, but I like the story and the one about the Old Man of the Mountain and the tunnels.'
'All right,' says the patient robot, eager to please, 'I'll tell you the tale as it was told to me. . .'

And he tells it, just as it was told to him. He weaves a story of the Crusades, throwing in dates and names, geographical references of a useful sort, and observations on the climate and the people, differentiating between what is known to be fact and what is romance or conjecture. Robot tells the story well; he may have been instructed in the art by Peter Ustinov, Richard Burton, or some other master handler of the language.

The childish audience may interrupt with questions or diversions to which the robot responds patiently and truthfully. ('I don't know' is one true response the robot can make.) Robot always returns to the narrative where it was interrupted unless instructed otherwise.

Robot doesn't know Saladin from a right triangle, but he can recite stories about selected subjects in logical order and great style. He can drill his master in multiplication tables and foreign language vocabulary (checking and commenting on pronunciation), explaining calculus concepts from different angles, and recite grammatical rules patiently until the kid can recite the same rules in his sleep.

Robot pet is an interactive system influenced by its master, but able to look after its own needs in critical ways as household pets do. The interests of the master will change with time and the information available from the idiot savant may be changed to suit the needs of the time. Perhaps information in ROM can be leased by the family for some period of time, then swapped for other material.

We may reasonably object that the store of information envisioned here cannot be carried in a household pet whose capacity is a mere one or two cubic feet largely occupied with motors, power supply, and other circuitry.

If we're talking about *household* pets, those that live in some limited area (a single building, a single neighborhood), we have opportunities in design that are not practical in other situations. Other robots may have to operate on remote battlefields or under the sea where communications lapses are likely. If your robot is operating on Mars and one-way communication takes twenty minutes, then everything necessary to the robot's well being must be aboard the vehicle. If the robot pet never strays farther than the backyard, though, it needn't carry its memory with it.

The mass memory system upon which the pet robot draws can easily be housed in a closet or hi-fi cabinet somewhere in the house. Magnetic pulse, radio, or modulated light communication between the robot and its memory can be accomplished in a tiny fraction of a microsecond and redundant channels can make that communication highly reliable. It matters little to your toes that your brain is way up there in your head. Critical matters like yanking your toe out of a fire are handled by local reflex circuitry without the immediate participation of the distant brain. The robot vehicle shambling around the house can similarly use its reflexes to handle matters of immediate concern. Only affairs that require some deliberation or a flow of information need involve mass memory at a distance.

It should be possible to store the 'character' of the robot pet in the central memory, to store the personality traits, the learned information about local terrain, voice and speech characteristics, all of the information that makes the robot pet one of the family. The 'physical body' of the robot would be, in effect, merely a puppet with mobility, reflexes, sensors, and cosmetic features. That body might be changed from time to time.

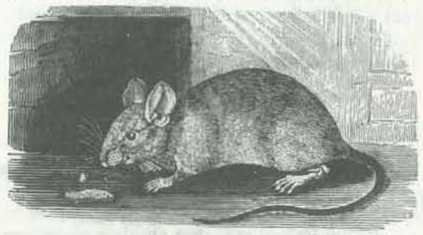
The vehicle might be upgraded, made stronger, given a better voice, new upholstery, greater or lesser size, hands, better vision. The determinant for this might be expense: as it goes along, the family might be able to afford improvements -- but who would wish to sell off or trade in the beloved family retainer? If the character of the robot remained in the permanent system, perhaps it would not be such an emotional wrench to change the vehicle, the vessel in which the character is embodied.

The basis for change might be a need for performance change. An arboreal robot could seem more useful than a swimmer. A husky outdoor pet might be needed instead of one that sleeps on your bed. The appearance of the creature could be changed to soothe a visiting relative who is made nervous by a gangly robot with a long reach, but is reassured by a jolly little round critter. The vehicles could be changed as one changes bits on a drill, to suit the present special need.

And there we are once again with the original concern of this discussion, the need to make robots acceptable and appropriate to the people and situation in which they must operate. There must be many solutions to this problem in addition to the few suggested here. ■

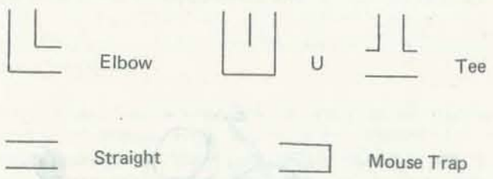
Micro-Mice Maze Contest

OK robotic fans, how about building a micro-based mouse? That can navigate mazes? **Spectrum**, *a publication of the IEEE is sponsoring a contest: you design and build a mouse, they design and build a maze. Plans are still in preliminary stages; we'll keep you informed as things develop. Or get yourself on Spectrum's mailing list by contacting: Howard Falk, Managing Editor,* **Spectrum**, *IEEE, Inc., 345 East 47th Street, New York, NY, 10017, (212) 644-7574.*

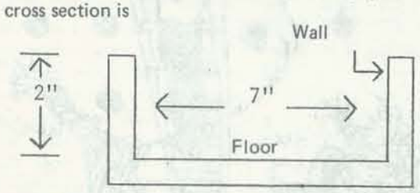


SPECTRUM'S PRELIMINARY SPECIFICATIONS

1) Viewed from inside the maze, the following, and only the following situations will be found (each of these may be oriented north, south, east, or west).



2) The maze is constructed of pine, its walls are painted white; the floor is paved with #00 sandpaper. The maze cross section is



3) No elephants or giraffes are allowed. All mice must be completely self-contained with no connection by wire or wireless to any outside equipment.

4) Shortest path from start to finish will be 217 inches.

QUESTIONS AND ANSWERS

Q. Can the mouse owner be present during coursing of the mice?
A. Owners will be expected to start their mice into the maze and recover them at the maze exit.

Q. Will guidance of mice by flashlights or ultrasonic sounders be allowed?
A. No. Mice must be completely self-contained.

Q. Will training runs be allowed for micro-mice?
A. There will be a prize for the best maze-running time after (2) preliminary runs. There will also be a prize for the fastest first-trial time.

Q. Will there be more than one possible path to the maze exit?
A. Yes.

Q. Will micro-mice be permitted to turn around, backup, move on wheels, legs, etc.?
A. Yes. Locomotion methods are at the option of the builders.

Q. Can mice be altered by external means once the runoffs begin?
A. No. All mice must be submitted and placed in official cages before the runoffs. No alteration (other than training) will be allowed. Any necessary repairs must be done under the supervision of an official referee.

PILOT

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. Ellen offers suggestions in how to write CAI in an article that begins on page 26.

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

```
Hi.  What's your name?
>Chris
 This program is about classification and stereotyping.
 Please give me some information.  What sex are you?
>female
 Thanks. What race are you?
>caucasian
 What's your cultural background?  (Italian? West Indian?)
>French
 O.K. I know you're a student.  What's your major,
 or your intended major?
>math
 All right.  One more question.  What's your religious  background?
>Protestent
 Fine. Now that I know you're a female, caucasian, French, and
 Protestent, how much do I know about you?
>Not much!
 Right.  I know very little.
 It seems to me that I just have a bunch of stereotypes in my head.

 Let's explore them.
 We all know about college students, right?   Pampered,
 unappreciative, sex in the dorms. . .

 Your turn.  Use two lines if you need them.  Everyone knows
 a female is . . .
>dependent, illogical, fickle
>vain, bad at math
 Now about the caucasian race, they're . . .
>colonizers, exploiters, controllers of the world's wealth
>dominated by males and technology
 The thing about Protestent people is that they . . .
>are smug, self-righteous
>want to impose their ways on others
 Do you know how math majors are stereotyped here at Stanford?
>no
 Go ask an upperclassman.

 All right, Chris:

      dependent, illogical, fickle
      vain, bad at math
      colonizers, exploiters, controllers of the world's wealth
      dominated by males and technology
      are smug, self-righteous
      want to impose their ways on others

 Does that sound like you?
>Nope
 You're much more complex than those stereotypes, yet, as you know,
 decisions are often based on such labelling.  Some landlords
 won't rent to students because, "They always wreck the place."
 Many jobs are closed to Blacks and Chicanos because,
 "They aren't dependable."

 Has anyone ever said to you,
 "Chris, you really shouldn't do that because you're a female?
>Oh yes
 You were probably pushing at your sex stereotype and
 upsetting people.

 Classification or categorization of people has functional value
 sometimes.  However, we need to stay aware of the difference
 between fact and inference, to be suspicious of our stereo-
 typical assumptions, and to realize that a person is described
 only in part, and perhaps inaccurately, by a label of any kind.

 Bye for now, Chris.  If you want to do a program on
 non-stereotypical thinking, try MUSTARD.  If you'd like to
 study fact and inference, do MARJORAM.

 Thanks.
```

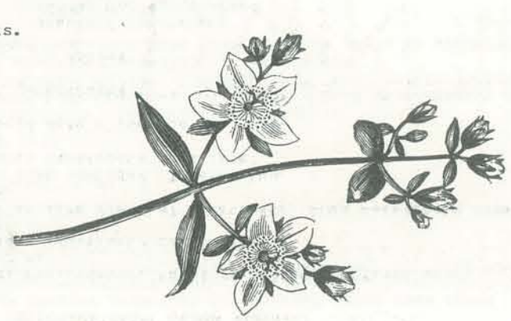
PROGRAMS BY ELLEN NOLD & SALLIE CANNOM

These programs are used in freshman English composition classes at Stanford University. They're written in PYLON, a close relative of PILOT. The main differences between PYLON and PILOT 73 (the 'standard' PILOT) are:

PYLON uses ¦ instead of commas to separate items in a match statement.
PYLON uses @ to identify variable, instead of $ as in PILOT 73 or \ as in Dean Dean Brown's Z-80 PILOT.
PYLON'S compute statement, C, allows access to capabilities of SPITBOL, a version of SNOBOL used at Stanford.

CUMIN

```
R: CUMIN by Ellen Nold and Sallie Cannom Stanford 8/73

T: Hi.  What's your name?
*Name A:
T: This program is about classification and stereotyping.
T: Please give me some information.  What sex are you?
*Sex A:
T: Thanks. What race are you?
*Race A:
T: What's your cultural background?  (Italian? West Indian?)
*Culture A:
T: O.K. I know you're a student.  What's your major,
T: or your intended major?
*Major A:
T: All right.  One more question.  What's your religious  background?
*Religion A:
T: Fine. Now that I know you're a @sex@, @race@, @culture@, and
T: @religion@, how much do I know about you?
A:
M: |NC|LIT|
T Y: Right.  I know very little.
T N: Really?
T: It seems to me that I just have a bunch of stereotypes in my head.
T:
T: Let's explore them.
T: We all know about college students, right?   Pampered,
T: unappreciative, sex in the dorms. . .
T:
T: Your turn.  Use two lines if you need them.  Everyone knows
T: a @sex@ is . . .
*Sex1 A:
*Sex2 A:
T: Now about the @race@ race, they're . . .
*Race1 A:
*Race2 A:
T: The thing about @religion@ people is that they . . .
*Religion1 A:
*Religion2 A:
T: Do you know how @major@ majors are stereotyped here at Stanford?
A:
M: |NO|UH U|
T Y: Go ask an upperclassman.
J Y: @go@
T N: O.K.  People say that @major@ majors are . . .
A:
A:
*Go T:
T: All right, @name@:
T:
T:      @sex1@
T:      @sex2@
T:      @race1@
T:      @race2@
T:      @religion1@
T:      @religion2@
T:

T: Does that sound like you?
A:
T: You're much more complex than those stereotypes, yet, as you know,
T: decisions are often based on such labelling.  Some landlords
T: won't rent to students because, "They always wreck the place."
T: Many jobs are closed to Blacks and Chicanos because,
T: "They aren't dependable."
T:
T: Has anyone ever said to you,
T: "@name@, you really shouldn't do that because you're a @sex@?
A:
M: |NO|UH U|NE|
T Y: Then either you've conformed closely to your sex stereotype,
T Y: or you've lived in a very permissive environment.
T N: You were probably pushing at your sex stereotype and
T N: upsetting people.
T:
T: Classification or categorization of people has functional value
T: sometimes.  However, we need to stay aware of the difference
T: between fact and inference, to be suspicious of our stereo-
T: typical assumptions, and to realize that a person is described
T: only in part, and perhaps inaccurately, by a label of any kind.
T:
T: Bye for now, @name@.  If you want to do a program on
T: non-stereotypical thinking, try MUSTARD.  If you'd like to
T: study fact and inference, do MARJORAM.
T:
T:
T:
T: Thanks.
T:
T:
END:
```



Traditional programming languages are often stumbling blocks to teachers and others who don't want to learn to program but want to use computers to teach particular subjects via instructional dialogues. Many people have found that PILOT is a way around the problem. The language is compact and powerful, well-suited for CAI which requires little computing facility. Even first graders have learned to write story-generating programs in PILOT.

Home and school microcomputers are proliferating at a rapid rate; now is the time to get easy-to-learn and easy-to-use languages onto cheap computers. *People's Computers* will continue to offer assistance to that end. We published source code in our Vol 5 No 5 for an experimental version of Z-80 PILOT by Dean Brown of Zilog. We've continued to report on revisions, and we'll let you know as soon as it's available on paper tape--it'll cost under $10.

John Starkweather, PILOT's originator, has completed an 8080 PILOT under contract from the National Library of Medicine. The user's manual and source code are in the April and May issues of our sister publication, Dr. Dobb's Journal.

8080 PILOT is available electrically by dialing in (the preferred mode), paper tape, ICOM and ISIS-II diskettes, and 9-track magnetic tape. Contact Bill Ford, National Library of Medicine, Lister Hill Center for Biomedical Communication, 8600 Rockville Pike, Bethesda, MD 20014. Lister Hill is also developing an extended PILOT for the NOVA/ECLIPSE family of computers.

Another source of information on PILOT is the PILOT Information Exchange, run by Gregory Yob, PO Box 354, Palo Alto, CA 94301. He maintains a library of materials in PILOT, source code in several languages, and a membership list with a brief note as to who is doing what.

◆ ◆ ◆

A brief summary of PILOT commands:

R: the Remark command indicates that the text which follows on the same line is a comment about what that part of the program does, and is not part of the program.

T: the Type command types out on the terminal the text which follows on the same line.

A: the Ask (or Answer or Accept) Command causes the computer to pause and wait for user input.

M: the Match command 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: the Jump command causes a branch to the label named: that command will be executed next.

Y: and N: the Yes and No commands test the match flag. 'JY:' means 'jump, if the match flag is yes, or turned on.' If the Y or N command occurs by itself, the 'Type' command is used. 'N:' means 'type this if the match flag is no, or turned off.'

E: the End command stops execution of that part of the program in which it occurs.

C: the Compute command, if available, permits evaluation of at least some arithmetic expressions.

U: the Use command calls subroutines.

SAGE

```
B: Sage by Ellen Nold and Sallie Cannom  Stanford 7/73

T: Hi! What's your name?
*Name A:
T:O.K.,@name@, this time we're going to talk about what
T:you can and can't see, hear, touch, taste and feel.
T:You just touched the computer terminal.  How does it feel?
*Feel A:
T:Fine.   Now say it again in a different way.
T:It feels like. . . .
*feel2 A:
T:Good.  Even more far out?  It feels as if. . . .
*Feel3 A:
T:You've just written a poem.  Here it is:
T:
T:
*T T:         TERMINAL
*E T:
*R T:         @feel@
*M T:         Like @feel2@
*I T:         Just as if @feel3@
T:
T:Do you want to change your poem?
A:
M: |NO|NOT|UH U|DONT|DON'T|
C Y: Set POEM1 to 0
J Y: @jump@
C:Set POEM1 to 1
T:O.K.  Use the next three lines to rework your poem.
T:
*P1Y A:
*P1E A:
*P1S A:
T:
T:That's the way!
*Jump T Y:I thought your poem was O.K., too.
T:
T:Now think about the things you normally can't see, hear
T:touch, taste or feel--like love, or hate, or joy or desperation.
T:Can you point to "desperation" the same way you can point to
T:a desperate man?
A:
M:|NO|NOT|DON'T|DONT|UH U|
T N:Well, you can describe desperation,
T N:but it would be hard to point to.
T Y:You're right.
T:A poet can touch an abstract idea like desperation
T:with words.  Desperation feels like. . .
A:
T:Yeah.  And desperation tastes like. . .
A:
T:Oh?  Desperation looks and smells. . .
A:
T:That was harder, but now we can see, smell, taste
T:and touch desperation.  Want to go on?
A:
M: |NO|NOT|
```

SAGE, Continued

```
J Y:@A1@
T N:Good. Now, @name@, think of another abstract idea.
*Abstract A:
T:What does @Abstract@ smell like?
*Smell A:
T:It smells like @smell@? Neat. What does @abstract@ sound like?
*Sound A:
T:Sounds like @sound@? Interesting.
T:What does @abstract@ look like?
*Look A:
T:You could make a poem out of that.
T:
*A T:     @abstract@
T:
T:          by @name@
*Z T:
*% T:       @smell@: @abstract@
*Ç T:     @sound@: Oh, @abstract@!
*V T:     @look@
T:
T:Now, @name@, would you like to take an imaginary
T:walk in the woods?
A:
M: |NO|NOT|DONT|DON'T|UH U|
T N:Good.  It's autumn, and as you're walking through the woods,
T N:what colors do you see?
J Y: @A2@
*Colors A:
T:What do @colors@ remind you of?
A:
T:Uh huh. Walking in the woods can lead to thoughts of abstract
T:ideas such as death, change, beauty or cycles.
T:Think of some more abstracts and type them in.
*More A:
T:How do you feel when you think about @More@?
A:
T:O.K., would you like to write a short poem, @name@?
A:
M: |NO|NOT|NEV&R|DONT|DON'T|
C Y: Set POEM2 to 0
J Y: @A1@
C: Set POEM2 to 1
T:O.K.  A suggestion is to type four lines
T:using the colors, @colors@,
T:and @more@, your abstract ideas,
T:and the feelings you have about @more@.
T:
*P2A T:          @name@'s Poem
*P2B T:
*P2C A:
*P2D A:
*P2E A:
*P2F A:
T:
T:Nice, @name@.
*A2 T:Here's something someone else wrote:
T:
T:       One by one,
T:       Branch by branch
T:       Leaves topple,
T:       Zigzag
```

```
T:       Through motionless October
T:       Struggle,
T:       Founder--
T:       Golden birds
T:       With broken wings.
T:
T:This poet makes the abstraction of "October" touchable or more
T:concrete by describing leaves.  To what does he compare leaves?
A:
M: |BIRD|FOWL|FEATH|
T Y:Yes, birds.
T N:Not exactly.  How about "birds"?
T:How do you think this poet felt about October?
A:
M: |SAD|DEP|SOM|CRY|HURT|PAIN|SOB|HELP|M&L|PEN|TR|
T Y:I agree.
T N:Really?  I didn't pick that up.
T:Were your feelings about the woods at all similar to this poet's?
A:
M: |NO|NOT|DONT|DON'T|
T N:You see that your feelings are touchable, too.
T Y:No?  Well, even so, you can see that
T Y:your feelings are touchable, too.
T:
T:Words can help us "touch" something abstract--like a feeling
T:or a season--though we can't hold it in our hands.
T:
*A1 T: Here, again are some of your words from this program:
T:
T: @T@
T: @E@
T: @R@
T: @M@
T: @I@
T:
T (POEM1=1): @P1Y@
T (POEM1=1): @P1E@
T (POEM1=1): @P1S@
T:
T: @A@
T: @Z@
T: @%@
T: @Q@
T: @V@
T:
T (POEM2=1): @P2C@
T (POEM2=1): @P2D@
T (POEM2=1): @P2E@
T (POEM2=1): @P2F@
T:
T:When you're ready to proceed, press return key.
A:
T:If you'd like to do more with abstract and concrete, try CLOVE.
T:Bye for now, @name@.
T:
T:
End:
```

CLOVE

```
R: CLOVE by Ellen Nold and Sallie Cannom  Stanford  8/73

T: Hi.  What's your name?
*Name A:
T: @name@, are you a man or a woman?  Type one word.
*Sex A:
T: Thanks.  Now we're ready to go.
T:
T: Language and most ordinary kinds of thought processes
T: are based on classification.
T: When I say "cat," what do you think of?
*Cat A:
T: You think of @cat@?  That's interesting.
T: I think of something four-legged, whiskery, and furry.
T: Close to what you pictured?
T: When I say "animal", what do you think of?
A:
T: I bet we're much farther apart on that.  I was thinking
T: of something bulbous, slimy, and sticky-tongued.
T:
T: "Animal" is a more general label than "cat," or conversely,
T: "cat" is more specific than "animal."
T: Can you add a word to "cat" to make it even more specific?
T: Type it in.
*Specific A:
T: You could say tabby cat, or my cat, or jungle cat, or any-
T: thing that further defines cat.
T:
T: Now type in a more general term for animal.
*General A:
T: That's harder.  What occurs to me is something like "living things."
T:
T:       @general@
T:       animal
T:       cat
T:       @specific@ cat
T:
T: You have done more than just classify "big" to "little," you
T: have  ordered a universe on four levels.
T: Add two more levels.  You may change your labels completely,
T: but make sure you have six levels - from most general to
T: most specific.  Use the next six lines.
T:
A:
A:
A:
A:
A:
A:
T:
T: By classifying those notions in your head, you have again
T: created a particular kind of universe.  You can destroy it and
T: create another simply by re-ordering those same ideas
T: in a different way.  Sound wierd?
A:
M: |NO|NOT|UH U|
T Y: Good.  You understand the influence language has on reality.
T N: Well, you perform this exercise daily.
T:
T: Of course, you're part of a universe too, @name@, . . .mine.
T:
T:          Galaxy 1501
T:          Solar System 10
T:          Earth
T:          United States
T:          California
T:          Stanford
T:          Stanford @sex@
T:          @name@
T:          @name@'s nose
T:
T: Like your place in my universe?
A:
T: Why?
A:
T: Fair enough.
T: You can make your universe to include or exclude
T: whatever you want.
T: Want to do your own?
A:
M: |NO|NOT|UH U|
J Y: @Q@
T N: Fine.  Use the next eight lines.
T:
A:
A:
A:
A:
A:
A:
A:
T:
*Go T:
T: Well @name@, we've just made some universes.
T: We do shape reality by our mental gyrations
T: and our choice of words.
T: Does that seem overstated?  If so, you might
T: want to pay further attention to the interrelationship
T: between language/thought/reality.
T:
T: The concepts of "abstract" and "concrete" are closely
T: related to what we've been doing here.  If this has been
T: interesting, try SAGE.  If you'd like to explore further
T: the idea of classification, try CUMIN.
T:
T: That's all for now, @name@.
T:
T:
END:
```

A COMPUTER IN YOUR POCKET FOR $145!

PROGRAMMING The HP-25

BY DAVID W. JOHNSTON

The HP-25 programmable calculator is the lowest priced member of the Hewlett-Packard programmable calculator line. It has 49 programming steps, eight data storage registers and eight conditional comparisons: a computer for $145. It's 'bang per buck' is hard to beat. This article is intended for the individual with little or no programming experience. An HP-25 with 'HP-25 Owners Handbook' should be available. This article should be considered a supplement to the handbook rather than a replacement for it.

An easy way to learn programming is to start with a simple equation such as the equation for a straight line, y = b + mx, where m is the slope and b is the y intercept. First, turn the HP-25 on, switch to PRGM and key in the following series of keystrokes.

Line	Key	Display	Remarks
01	RCL 3	24 03	Recall x in REG 3
02	RCL 2	24 02	Recall m in REG 2
03	X	61	Product of m and x
04	RCL 1	24 01	Recall b in REG 1
05	+	51	Add b to mx for y
06	R/S	74	Display y
07	GTO 01	13 01	Return to program start

Switch to RUN and then f PRGM to beginning of program. Store b in register 1, m in register 2 and x in register 3. Press R/S to run the program. For x = 3, m = 2 and b = 1, y = 7. For x = 4, m = 3 and b = 2, y = 14. For all values on the right hand side of the equation equal to 10, y = 110. Try these values out to see if the program has been entered correctly.

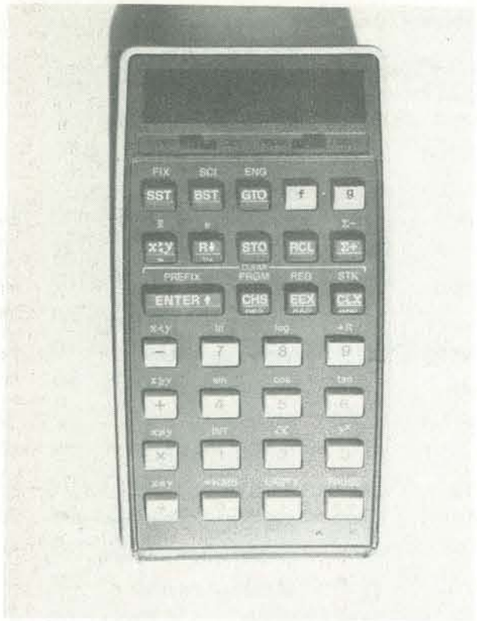
M and b are constants and usually do not need to be changed too often. The above program may be rewritten with the values for m and b in the program. Let m = 3 and b = 2; the program is as follows.

Line	Key	Display	Remarks
01	RCL 3	24 03	Recall x in REG 3
02	3	03	Key in 3
03	X	61	Product of 3 and x
04	2	02	Key in 2
05	+	51	Add 2 to 3x for y
06	R/S	74	Display y
07	GTO 01	13 01	Return to program start

As before, switch to RUN and f PRGM. Store x in register 3 and press R/S to run the program. It is possible to simplify the program by replacing line 01 with NOP and keying in the x value before pressing R/S.

The above program can be expanded so that x values are automatically generated, starting with 0 and increasing in increments of 1 to as large as desired. The program is as follows.

Line	Key	Display	Remarks
01	0	00	Key in 0
02	STO 3	23 03	Store in REG 3
03	RCL 3	24 03	Recall x in REG 3
04	PAUSE	14 74	Display x for a second
05	3	03	Key in 3
06	X	61	Product of 3 and x
07	2	02	Key in 2
08	+	51	Add 2 to 3x for y
09	PAUSE	14 74	Display y for a second
10	1	01	Key in 1
11	STO + 3	23 51 03	Store 1 in REG 3, adding to contents
12	GTO 03	13 03	Go to line 03



HP-25

This equation is rather simple and can be solved without programming, but for more than several solutions programming is more efficient. The sequence of keys is shown below.

x - key in x-value

ENTER - move x-value up in stack

m - key in m-value

X - product of x and m

b - key in b-value

+ - add product and b-value

It is not as easy to solve a more complex equation. The key sequence given below is for the equation $y = c + bx + az^2$.

z - key in z-value

gx^2 - square z-value

a - key in a-value

X - product of z^2 and a

x - key in x-value

ENTER - move product, x-value up in stack

b - key in b-value

X - product of x and b

+ - add az^2 and bx

c - key in c-value

+ - add c to $bx + az^2$

This is a rather cumbersome calculation if this calculation is to be done a number of times. It would be simpler to use a program like that given below. There are two different versions, one includes the constants in the program and the other uses registers for the constants. It is also possible to generate the x-value automatically as was done before in the equation y = b + mx.

First, the program to calculate $y = c + bx + az^2$ with the constants in the program is shown.

Line	Key	Display	Remarks
01	5	05	Key in b, 5 in this case
02	RCL 3	24 03	Recall x in REG 3
03	X	61	Multiply 5 by x
04	STO 6	23 06	Store product in REG 6
05	RCL 5	24 05	Recall z in REG 5
06	RCL 5	24 05	Recall z in REG 5, again
07	X	61	Square of z
08	3	03	Key in a, 3 in this case
09	X	61	Multiply 3 by z^2
10	RCL 6	24 06	Recall REG 6
11	+	51	Add contents of REG 6 to $3z^2$
12	1	01	Key in c, first 1
13	0	00	then 0, to make 10
14	+	51	Add 10 to obtain y
15	R/S	74	Display y
16	GTO 01	13 01	Return to program start

To use this program press f PRGM, store x in REG 3, store z in REG 5 and press R/S. For x = 4 and z = 2, you should get y = 42; for x = 1 and z = 1, you should get y = 18. If x = 10 and z = 10, then y = 360. The above program may be

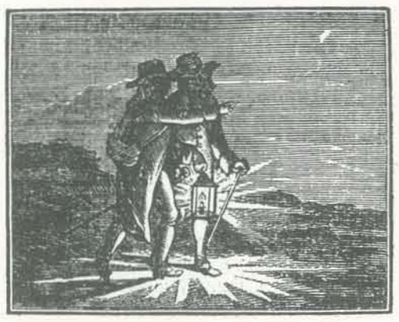
modified to enter x and z by using the R/S key as shown below.

Line	Key	Display	Remarks
01	5	05	Key in b, 5 in this case
02	R/S	74	Stop program, key in x
03	X	61	Multiply 5 and x
04	STO 6	23 06	Store product in REG 6
05	R/S	74	Stop program, key in z
06	gx^2	15 02	Square z
07	3	03	Key in a, 3 in this case
08	X	61	Multiply 3 by z^2
09	RCL 6	24 06	Recall REG 6
10	+	51	Add contents of REG 6 to $3z^2$
11	1	01	Key in c,
12	0	00	10 in this case
13	+	51	Add 10 to obtain y
14	R/S	74	Display y
15	GTO 01	13 01	Return to program start

To use this program press f PRGM. Press R/S, then key in x. Press R/S, then key in z. Press R/S to ready program for next x and z values. Test program by using values given for program using REG.

If the constants, a, b, and c, change quite a bit, it would be more convenient to store them in registers as well as the variables. Another way would be to store the constants in registers and read the variables with the R/S key. It would also be possible to read in both constants and variables with the R/S key, but this would be quite a few entries per program run. It would be possible to generate x or z automatically as was done with the equation $y = b + mx$.

With the above background, it is possible to consider a more complicated equation such as $y = c + bx + az^2 + dw^3$. The program may be written as follows.



Line	Key	Display	Remarks
01	3	03	Key in d, 3 in this case
02	ENTER	31	Move 3 up in stack
03	R/S	74	Stop program, key in w
04	ENTER	31	Move w up in stack
05	3	03	Key in exponent 3
06	fy^x	14 03	Raise w to 3rd power
07	X	61	Multiply w^3 by 3
08	STO 1	23 01	Store in REG 1
09	5	05	Key in a, 5 in this case
10	ENTER	31	Move a up in stack
11	R/S	74	Stop program, key in z
12	gx^2	15 02	Square z
13	X	61	Multiply 5 times z^2
14	STO + 1	23 51 01	Add product to REG 1
15	7	07	Key in b, 7 in this case
16	R/S	74	Stop program, key in x
17	X	61	Multiply 7 times x
18	STO + 1	23 51 01	Add product to REG 1
19	RCL 1	24 01	Recall REG 1
20	1	01	Key in c, first 1
21	0	00	then 0, to make 10
22	+	51	Add contents of REG 1 to 10
23	R/S	74	Stop program, display y
24	GTO 01	13 01	Return to program start

In order to use this program, press f PRGM; then press R/S and key in w (Note: w must be greater than 0 or an Error display will result). Now, press R/S and key in z. Next, press R/S and key in x. Lastly, press R/S to obtain value of y. The above program will not run if w is equal to or less than 0. The first 7 steps may be changed, as shown below, so it will run with any value of w.

Line	Key	Display	Remarks
01	3	03	Key in d, 3 in this case
02	ENTER	31	Move 3 up in stack
03	R/S	74	Stop program, key in w
04	ENTER	31	Move w up in stack
05	gx^2	15 02	Square w
06	X	61	Cube w
07	X	61	Multiply w^3 by 3

The rest of the program is as before and it is run as before.

The program may be changed so that the constants, a, b, c and d, are entered by register instead of being part of the program. It may be written as follows.

Line	Key	Display	Remarks
01	RCL 1	24 01	Recall d in REG 1
02	ENTER	31	Move d up in stack
03	R/S	74	Key in w
04	ENTER	31	Move w up in stack
05	gx^2	15 02	Square w
06	X	61	Multiply w^2 by w
07	X	61	Multiply w^3 by d
08	STO 6	23 06	Store dw^3 in REG 6
09	RCL 2	24 02	Recall a in REG 2
10	ENTER	31	Move a up in stack
11	R/S	74	Key in z
12	gx^2	15 02	Square z
13	X	61	Multiply z^2 by a
14	STO + 6	23 51 06	Add az^2 to REG 1
15	RCL 3	24 03	Recall b in REG 3
16	R/S	74	Key in x
17	X	61	Multiply x by b
18	STO + 6	23 51 06	Add bx to REG 1
19	RCL 6	24 06	Recall REG 6
20	RCL 4	24 04	Recall c in REG 4
21	+	51	Add contents of REGs 6 and 4
22	R/S	74	Display y
23	GTO 01	13 01	Return to program start

The program may be used in the following manner. Store d, a, b and c in registers 1, 2, 3 and 4 respectively. Press f PRGM and then R/S. Next, key in w, press R/S. Then, key in z and press R/S. Key in x and press R/S to obtain value of y.

The above examples show some of the ways that the HP-25 can be programmed. They are by no means comprehensive, but it is hoped that they will prove helpful to those who wish to develop their programming skills. As stated before, an HP-25 with 'HP-25 Owner's Handbook' should be available to get the most benefit from this article.

Last minute update: Hewlett Packard has announced their HP-25c with a new type of memory, making possible the storage of programs for several weels or more. Its price is $200. ■

SR-56 PROGRAMS

These programs for the SR-56 calculator are in the public domain: you just pay copy costs, 5¢ per page, and supply a SASE (self-addressed stamped envelope). Be sure to provide an adequately large SASE for the number of sheets you order. The postage of the SASE should be computed at four Xerox sheets for one oz. It would be 13¢ for 1-4 sheets, 24¢ for 5-8 sheets, 35¢ for 9-12 sheets, etc. To obtain the number of sheets divide the cost by 5, thus a 15¢ program would have three sheets.

To order programs send a self-addressed stamped envelope with price of programs; stamps accepted, please make the 13¢ or less, to: David W. Johnston
P.O. Box 3781
Washington, D.C. 20007

Submitted programs should be Xerox reproducible and should contain: formula or equation with necessary description; program listing including line number, key, display and remarks; user instructions with solved example and author's name and address if someone wants more information on the program. The author is the best expert on the program. A copy should be sent to David W. Johnston at the above address and also to:
Richard C. Vanderburgh
SR-52 USERS CLUB
9459 Taylorsville Road
Dayton, OH 45424

MATHEMATICS
1 20-Digit Product 5¢
2 Real Number Factorial 5¢
3 Three Simultaneous Equations 5¢
4 4 x 4 Determinant 5¢
5 3 Linear Equations in 3 Unkowns 5¢
6 N! accuracy for≤999 5¢
7 Factors of Integers (three programs) 20¢
8 Area of a General Quadrilateral 10¢
9 Polar to Rectangular, Bolt Circle (three programs) 15¢
10 Automatic Looping Routine 5¢
11 Automatic Factorial Routine 5¢
12 2 x 2 Matrix Determinant and Inverse 10¢
13 2 x 2 Matrix Add., Sub., and Mult. 15¢
14 Translation and Rotation of Coordinates 10¢
15 3 x 3 Determinant 10¢
16 Hyperbolic and Inverse Hyperbolic Functions 15¢
17 Triangle Solution, Given Three Sides 15¢
18 Newton-Raphson Method for Equation Roots 10¢
19 Solution of a General Equation 5¢
20 IBM 360/370 Floating Point 5¢
21 Polygon Area Programs 5¢

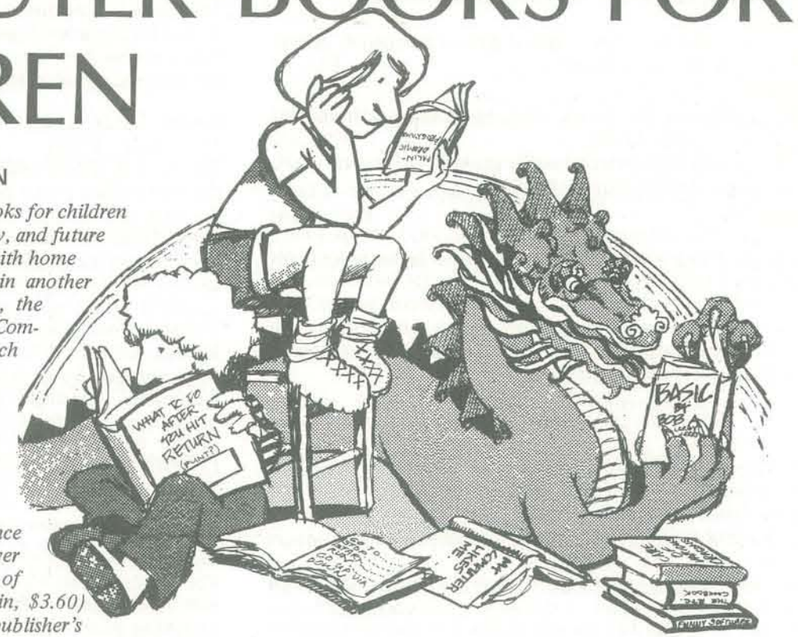
STATISTICS
22 Multiple Correlation (3 Variables) Coefficients 15¢
23 Linear Partial Correlation (3 Variables) Coefficients 15¢
24 Linear Partial Corr. (4 Var., 1 Const., 1 Ignor.) 10¢
25 Reliability or Probability of Success of Component 5¢

PHYSICS
26 Lens Maker's Equation (Thin and Thick Lenses) 10¢
27 Lorentz-Einstein Transformation Equations 10¢
28 Spectral Series of Excited Hydrogen Atoms 5¢
29 Hyperfocal Distance; Near, Far Depth of Field 10¢
30 Projectile Motion Equations 10¢
31 Current in Series AC Circuit 10¢
32 Column Design Program 10¢
33 Planck's Radiation Law 10¢
34 Venturi Flowmeter 10¢
35 Number of Atoms in Cube and Sphere 10¢
36 Capillary Rise in Tube 5¢
37 Probability of a Missile Hit 10¢
38 Radioactive Decay of Mother and Daughter Elements 10¢
39 Illumination in Image on Film 10¢
40 Several Thin Lens Focal Length Calculations 10¢
41 Program for Nodal Point Distances 10¢
42 Lens Angle of View and Illumination at Angle 10¢
43 Rocket Acceleration, Velocity and Distance 10¢
44 Program for Calculation of Orbital Velocity 10¢
45 Center of Mass Calculation for N Particles 10¢
46 Thermal Expansion; Linear, Area, Volume Coefficients 10¢
47 Program for Simple Harmonic Motion 10¢
48 Temperature Conversion Program 10¢
48A Thermal and Pressure Equilibrium of a Gas 10¢

GAMES
49 Nim B 10¢
50 Sun-Tsu's Trick 10¢
51 Game of Nimb 10¢
52 Automatic Strategy Dynamic Nimb 5¢
53 Twenty-One "A" 5¢
54 Hi/Lo (0–1000), Guess Your Age in 5 Guesses 5¢
55 Pinochle Score Keeper 5¢

MISCELLANEOUS
56 Biorhythm 5¢
57 Biorhythm II 5¢
58 Biorhythm III 10¢
59 Daily-compounded Passbook Savings Account 5¢
60 Shopping Program for Taxable and Non-Taxable Items 5¢
61 Miles per Hour of Racing Car 5¢

COMPUTER BOOKS FOR CHILDREN



BY BETSY ROSEN

This overview of computer literacy books for children brings together an array of fact, fantasy, and future views for your perusal. Those of you with home computer systems may be interested in another type of computer book for children, the how-to-program book. The People's Computer Company Bookstore offers 'Teach Yourself BASIC', Volumes 1 & 2, by Bob Albrecht, $2.95 each, for kids 10 and up; 'My Computer Likes Me When I Speak in BASIC', by Bob Albrecht, $2.00, for kids 11 and up; 'BASIC', by Albrecht, Finkel, and Brown, $4.95, for kids 14 and up. And we hear that Bob Kahn at Lawrence Hall of Science is recommending Dwyer and Kaufman's book 'A Guided Tour of BASIC Programming' (Houghton Mifflin, $3.60) for kids as young as 8, although the publisher's description suggests completion of the first year of junior high school math as a prerequisite!

Computers are unavoidable today: they're in schools, banks, consumer markets, and even libraries. So the time has come to talk of computer literacy for people of all ages. For the novice, computer literacy is having a basic knowledge of simple concepts and terminology. The aim of such literacy is to provide the novice with a feeling of understanding when confronted with an 'expert' explanation.

Computer literacy at the novice level is needed by children as well as adults. Children learn a great deal from early introduction to subjects via fiction and non-fiction. A sense of familiarity with a subject gives a child confidence in pursuing it, and the feeling that this is an area in which he 'belongs'. Therefore, computer literacy should begin before the first course offered in the school system.

Several dozen books about computers have been written for children and are available at this time. These are of varying quality due to pressure on publishers to satisfy a current market, and the consequent need to produce 'instant' writers in the field. More such books are constantly appearing.

My search for such books began with a look at the 1976 *Children's Books in Print: Subject Guide* under the headings Automatic Data Processing, Automation, Automation-Fiction, Computers, Computers-Fiction, and Programming (Electronic Computers). I then added any listing under these headings from the three children's collections in the Champaign/Urbana (Illinois) area. Finally, I found twelve titles in these libraries which I could preview personally and I added to my list from the bibliographies they included.

In reviewing these materials I attempted to look at the quality of the book for its intended audience and to see if it might be used with less sophisticated readers among the adult population. Therefore, I was looking for the presence of absence of technical jargon which is often used in the more sophisticated materials in any field, and for clear simple explanations with good accompanying illustrations. I looked at format, style, organization and scope with the idea of their effect on the potential audience. Formal structure such as chapters, an index, and appendices are an aid to some and a barrier to others. The author's credentials and the accuracy of the material included were of interest to me. I was looking for whether the author stated his biases toward his subject or mentioned any of the potential issues in the field, such as problems of computer privacy. Finally, I looked at style and readability to see if the book contained difficult discussions of simple material.

What follows is an annotated bibliography of the twelve titles which I saw and a listing of those which I identified but was unable to preview.

ANNOTATED BIBLIOGRAPHY
COMPUTER LITERACY BEFORE HIGH SCHOOL

*Recommended books are marked by an asterisk, *. Prices, when available, are 1976 prices.*

*Berger, Melvin. *Computers.* illus. by Arthur Schaffert. Coward, 1972. 44p. $4.49.
A better simplistic presentation for grades K-3. It covers input, output, control and memory briefly.

*_____. *Those Amazing Computers!* Day, 1973. 189p. $5.95.
Illustrated with photographs and organized by uses, this book includes a table of contents, a bibliography for further reading and an index. It includes brief material on input, output, control unit, programming and flow charts, central processor, memory and data banks. A section mentions the dangers of misuse of computer privacy.

Corliss, William R. *Computers.* U.S. Atomic Energy Commision, 1966. 56p.
This summary with bibliography for further reading would be a starting point for a serious student. It includes problems and more detailed technical information in a more concise format then any of the other titles listed. For grades 7 and up.

*De Rossi, Claude. *Computers: tools for today.* Children's Press, 1972. 87p. $6.60
Written for grades 4-6, the text is clear and gives simple information about binary addition, bits, punched cards, card readers, magnetic tape, programming, programmers, flowcharts, and a little history. A table of contents and index are included.

Halacy, D.S., Jr. *Computers: The Machines We Think With.* Harper and Row, 1969. 279p. $9.95.
For the computer buff, this title (for grades 7 and up) is a discussion of the computer and its place in society which relates the computer to literature of the past and present as well as presenting basic information about the computer's history and possible future. It has a table of contents and an index.

Kenyon, Raymond G. *I Can Learn About Calculators And Computers.* Harper and Row, 1961. 112p. $5.49.
This book is really a 'how to build your own' and includes much more about history and 'how to' than it does information about computers. It includes a table of contents, glossary, and an appendix of sources for simple kits. Recommended for grades 5 and up, the instructions seemed to be plausible for that age.

* Meadow, Charles. *The Story of Computers.* illus. Anne Lewis. Harvey House, 1970. 124p. $5.89.
Recommended for grades 5-8; a glossary, index, bibliography and table of contents are included. I felt that De Rossi was simpler, clearer and contained more information.

Ray, Jo Anne. *Careers in Computers.* Learner, 1973. lv(mp). $3.95.
A career book with photographic format, written for K-4 and including sex stereotypes and no information relating to preparation for the various jobs.

Rusch, Richard B. *Man's Marvelous Computer: The Next Quarter Century.* Simon and Schuster, 1970. 128p.
This book is a discussion of the issues which computers will work with in the next quarter century; only a brief introduction describes how they work. The book includes a table of contents and an index. Recommended for grades 5 and up.

Synder, Gerald S. *Let's Talk About Computers.* J. David, 1973. 112p. $5.95;
The book is recommended only as supplemental reading for the intermediate grades because the format was confusing. Italics were used both for words not being defined and for words being defined. 'Bits' were defined and the pronounciation given was 'bytes'. Organized by uses, the table of contents and glossary appear adequate to the book.

*Srivastava, Jane Jonas. *Computers.* illus. by James and Ruth McCrea. Crowell, 1972. 32p. $4.50.
Recommended for grades 1-4, this short book includes a lot of information very simply presented. Input unit, punched cards, paper tapes, records, plastic tapes, oral input, program instructions, programmers, languages (ALGOL, FORTRAN, COBOL), arithmetic unit, memory unit bank, output unit, control unit, flowcharts plus a little history and some uses are presented.

Steinberg, Fred. *Computers.* Franklin Watts, Inc. 1969. 89p.
For the intermediate grades, this book is a photographic essay with its emphasis on what the computer does. A little dated in presenting punched cards as *the* input method and in the looks of the book. However the pages are sprinkled with both men and women (Only men in the real computer dealings, women where the public and computer meet). It includes a table of contents and an index.

BIBLIOGRAPHY

Recommended grade levels were provided by the publishers.

Adler, Irving. *Thinking Machines.* Rev. Ed. Day, 1973

Arnold, Pauline and Percival White. *Automation Age.* Holiday, 1963.

Baker, Eugene. *I Want To Be A Computer Operator.* Children's Press, 1973. (gr. K-3)

Berger, Melvin. *Computers.* Coward, 1972. 44p. (gr. K-3)

———. *Those Amazing Computers!* Day, 1973. 189p. (gr. 5-9)

Bernstein, Jeremy. *The Analytical Engine.* Random House, 1966.

Bowles, Edmund A. *Computers in Humanistic Research.* Prentice Hall, 1967.

Boy Scouts of America. *Computers.* BSA, 1973. (gr. 6-12)

Braude, Michael. *Larry Learns About Computers.* Denison, n.d.

Bruton, Erick and Langdon Goodman. *Automation.* Soccer, n.d.

Burck, G. "The Boundless Age of the Computer" *Fortune,* 69:101 (March 1964) Part one of four parts.

Corliss, William R. *Computers.* U.S. Atomic Energy Commission, 1966. 56p.

Cohen, L. *The Human Side of Computers,* New Ed. McGraw, 1975 (gr. 7-12)

Cook, Joseph. *Electronic Brain: How It Works.* Putnam, 1969. (gr. 5 and up)

Davis, H.M. "Mathematical Machines" *Scientific American,* 180:28 (April 1949)

DeRossi, Clause. *Computers: Tools For Today.* Children's Press, 1972. 87p. (gr. 4-8)

Dilson, Jesse. *Curves and Automation.* Soccer, 1971. (gr. 7-9)

Desmonde, William H. *Computers and Their Uses.* Prentice-Hall, 1964.

Englehardt, S.L. *Computers.* Pyramid Pubs., 1962.

Franke, Hubert W. *Computer Graphics; Computer Art.* Phaidon, 1971.

Greenberger, M. "The Computers of Tomorrow" *The Atlantic.* 213:63 (May 1964)

Halacy, Dan. *Charles Babbage: Father of the Computer.* MacMillan, 1970. (gr. 7-12)

– – – –. *Computers: The Machines We Think With.* Harper and Row, 1969. 27p. (gr.7 and up)

– – – –. *What Makes A Computer Work?* Little, 1973. (gr.3-5)

Holtzman, Wayne H. *Computer-Assisted Instruction.* Harper and Row, 1970.

Jones, Weyman. *Computers: The Mind Stretcher.* Dial, 1969.

Kenyon, Raymond G. *I Can Learn About Calculators and Computers.* Harper and Row, 1961. 112p. (gr. 4-6)

Kohn, Bernice. *Computers at Your Service.* Prentice-Hall, 1962. (gr. 3-6)

Leed, Jacob. *The Computer and Literary Style.* Kent State University, 1966.

Lewis, Alfred. *New World of Computers.* Dodd, 1965. (gr. 3-9)

Lincoln, Harry. *The Computer and Music.* Cornell University, 1970.

Loehlin, John C. *Computer Models of Personality.* Random House, 1968.

Lohberg, Rolf and Theo Lutz. *Computers at Work.* Stirling, 1969.

Meadow, Charles. *The Story of Computers.* Harvey House, 1970. 124p. (gr. 5-8)

Morrisin, Phillip and Emily. eds. *Charles Babbage and His Calculating Engines.* Dover Pubs., 1961.

Penrose, L.S. "Self-Reproducing Machines" *Scientific American,* 200:105 (June 1959)

Pfeiffer, John. *The Thinking Machine.* Lippincott, 1962.

Pierce, J.R. "How Smart Are Computers?" *Saturday Evening Post,* 234:24 (November 4, 1961)

Piper, Robert. *Story of Computers.* Harcourt Brace, 1964. (gr. 7 and up)

Ray, Jo Anne. *Careers in Computers.* Learner, 1973. lv. (unp) (gr K-4)

Reichardt, Jasia. *The Computer in Art.* Van Nostrand-Reinholdt, 1971.

Rice, Jean. *My Friend the Computer.* Houghton Mifflin, 1976

Rusch, Richard B. *Computers: Their History and How They Work.* Simon and Schuster, 1969. (gr.7 and up)

———*Man's Marvelous Computer: The Next Quarter Century.* Simon and Schuster, 1970. 128p. (gr. 5 and up)

Sanders, Donald H. *Computers in Business.* McGraw-Hill, 1972.

Seldin, Joel. *The Challenge of Men and Machines.* rev. ed. Coward, 1971. (gr.6-8)

Seligsohn, I.J. *Your Career in Computer Programming.* Messner, 1967. (gr.7 and up)

Snyder, Gerald S. *Let's Talk About Computers.* Jonathan David, 1973. 122p. (gr. 3-6)

Continued on page 58 . . .

CAPTURE

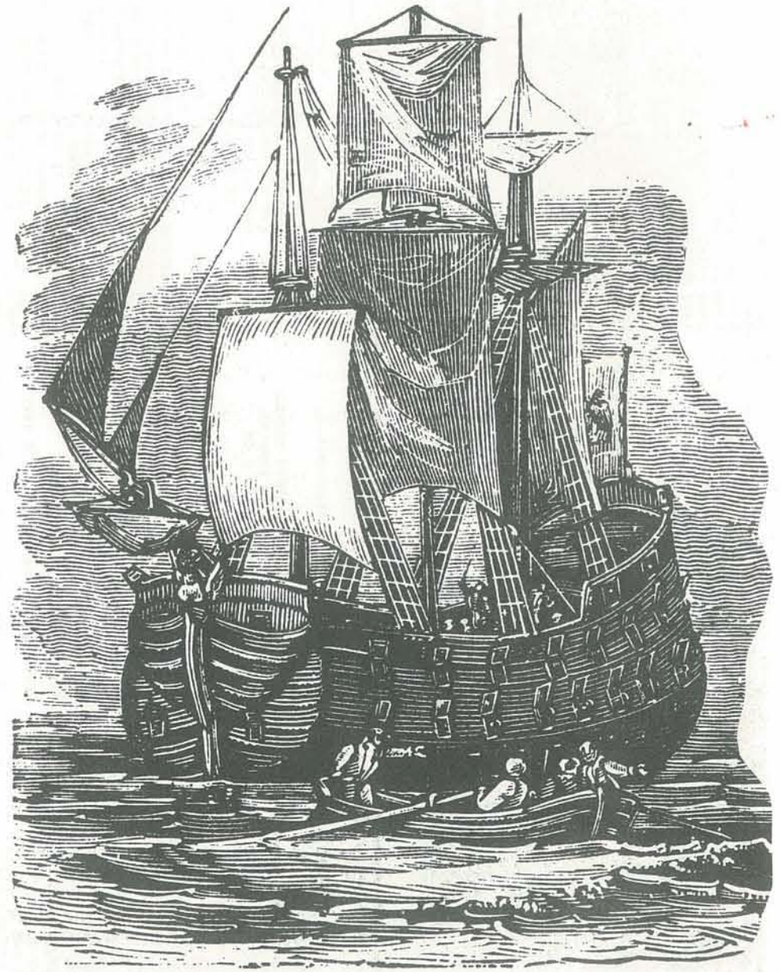
BY MAC OGLESBY

CAPTURE is a game which needs a computer. Its help is essential to even set up the playing board. Since the same beginning configuration wouldn't likely appear again in a million lifetimes, the replay option allows the user to go back and try again with the same set up. But don't count on the program making the same moves, even if you do. The user of CAPTURE gets involved with the concepts of randomness, density, estimation, metric measure and strategy.

```
WANT INSTRUCTIONS FOR CAPTURE? YES, PLEASE

HERE'S THE BOARD AT THE START:

  +                 H
       M  Y  G
Z  T  AX            O
         +       U
            Q             N
   +        B             +
 +       C  I
           V          ++
              K LW P
 +  +       +     RD
                        +
 FE J                   S

GOING IN TURN, THE PLAYERS (# AND &) CAPTURE ANY LETTER ON
THE BOARD.  NOT COUNTING SPACES, ALL CHARACTERS WITHIN ONE
CENTIMETER OF THE CAPTURED LETTER ARE ALSO CAPTURED AND
CHANGE TO THAT PLAYER'S SYMBOL.

THE GAME ENDS AFTER FOUR TURNS OR IF ALL LETTERS HAVE BEEN
CAPTURED.  THE PLAYER WITH THE MOST CAPTIVES WINS.

TO CAPTURE, TYPE ANY SINGLE LETTER SHOWN ON THE BOARD.
TYPE 'STOP' TO STOP OR 'RESIGN' TO RESIGN.

HOW MANY HUMAN PLAYERS  (1 OR 2)? 1
  OK, THE COMPUTER WILL PLAY THE #'S.
WHO GOES FIRST  (1=COMPUTER  2=YOU)? 2

HERE WE GO...
```



```
THE &'S CAPTURE? X
THE #'S CAPTURE P

  +                 H
       &  &  G
Z  &  &&            O
         &       U
            Q             N
   +        B             +
 +       C  I
           V          #+
              K ## #
 +  +       +     ##
                        +
 FE J                   S

THE &'S CAPTURE? I
THE #'S CAPTURE E

  +                 H
       &  &  G
Z  &  &&            O
         &       U
            &             N
   +        &             +
 +       &  &
           &          #+
              & ## #
 #  #       +     ##
                        +
 ## #                   S

THE &'S CAPTURE? O
THE #'S CAPTURE Z

  #                 &
       &  &  G
#  #  &&             &
         &       &
            &             N
   +        &             +
 +       &  &
           &          #+
              & ## #
 #  #       +     ##
                        +
 ## #                   S

THE &'S CAPTURE? N
THE #'S CAPTURE G

  #                 &
       &  #  #
#  #  &&             &
         &       &
            &             &
   +        &             &
 +       &  &
           &          #+
              & ## #
 #  #       +     ##
                        +
 ## #                   S
```



```
*** THE #'S WIN!!
THE #'S HAVE 16 CAPTIVES AND THE &'S HAVE 15.

WANT TO PLAY AGAIN? YES
```

capture

```
100 ' NAME:  ELEMLIB***:CAPTURE
110 '
120 ' BY:  MAC OGLESBY ON 05/29/76.
130 '
140 ' DESCRIPTION:  GOING IN TURN, THE PLAYERS (# AND &) CAPTURE
150 '   LETTERS PRINTED ON A BOARD OF 12 X 25 CELLS.  ALL CHARACTERS
160 '   (EXCEPT SPACES) WITHIN ONE CENTIMETER OF THE CAPTURED LETTER
170 '   ARE ALSO CAPTURED AND CHANGE TO THAT PLAYER'S SYMBOL.
180 '   THE PLAYER WHO ENDS UP WITH THE MOST CAPTIVES WINS.
190 '
200 ' INSTRUCTIONS:  TYPE "RUN" FOR COMPLETE INSTRUCTIONS.
210 '   NOTE:  THIS PROGRAM IS WRITTEN FOR TERMINALS WHICH PRINT
220 '   SIX LINES OR TEN SPACES PER INCH.
230 '
240 '
1000 DIM D(12,25),E(12,25),R(26),C(26),M(26),W(33),N(33)
1010 RANDOMIZE
1020 FOR J=1 TO 33                      'W() AND N() STORE CHANGES NEEDED
1030    READ W(J),N(J)                  '  TO ROW AND COLUMN TO EXAMINE
1040 NEXT J                             '  ALL CHARACTERS WITHIN 1 CM
1050    DATA -2,-2,-2,-1,-2,0,-2,1,-2,2
1060    DATA -1,-3,-1,-2,-1,-1,-1,0,-1,1,-1,2,-1,3
1070    DATA 0,-4,0,-3,0,-2,0,-1,0,0,0,1,0,2,0,3,0,4
1080    DATA 1,-3,1,-2,1,-1,1,0,1,1,1,2,1,3
1090    DATA 2,-2,2,-1,2,0,2,1,2,2
1100 LET P$(1)="#"                      'PLAYER SYMBOLS
1110 LET P$(2)="&"
1120
1130 PRINT "WANT INSTRUCTIONS FOR CAPTURE";
1140 LINPUT A$
1150 GOSUB 2730                         'PROCESS INPUT
1160
1170 'BOARD STARTS AS ALL SPACES
1180 FOR J=1 TO 12                      'NEW SETUP REPLAY RETURNS TO HERE
1190    FOR K=1 TO 25
1200       LET D(J,K)=32                'D(,) STORES ASCII VALUES FOR BOARD
1210    NEXT K
1220 NEXT J
1230 'PUT 26 LETTERS ON THE BOARD AT RANDOM
1240 FOR J=1 TO 26
1250    LET R1=1+INT(RND*12)            'TRIAL ROW
1260    LET C1=1+INT(RND*25)            'TRIAL COLUMN
1270    IF D(R1,C1)=32 THEN 1290        'SEE IF SPACE IS OCCUPIED
1280       GOTO 1250                    'TRY AGAIN
1290    LET D(R1,C1)=64+J               'PUT LETTER ON BOARD
1300    LET R(J)=R1                     'STORE EACH LETTER'S ROW
1310    LET C(J)=C1                     'AND COLUMN
1320 NEXT J
1330 'LET'S PUT SOME +'S ON THE BOARD, TOO.
1340 LET N1=10+INT(RND*91)              'FROM 10 TO 100 +'S
1350 FOR J=1 TO N1
1360    LET R1=1+INT(RND*12)
1370    LET C1=1+INT(RND*25)
1380    IF D(R1,C1)=32 THEN 1400
1390       GOTO 1360
1400    LET D(R1,C1)=43
1410 NEXT J
1420 MAT E=D                            'SAVE COPY OF BOARD FOR REPLAY
1430
1440 LET H1=H2=M=0                      'SAME SETUP REPLAY RETURNS HERE
1450 LET T=2                            'DEFAULT; #'S GO FIRST
1460 PRINT
1470 PRINT "HERE'S THE BOARD AT THE START:"
1480 GOSUB 2890
1490 IF SEG$(A$,1,1)<>"Y" THEN 1510
1500    GOSUB 3040                      'PRINT INSTRUCTIONS
1510 PRINT "HOW MANY HUMAN PLAYERS  (1 OR 2)";
1520 INPUT P
1530 IF (P-1)*(P-2)=0 THEN 1560
1540    PRINT "PLEASE TYPE 1 OR 2 ";
1550    GOTO 1520
1560 IF P=2 THEN 1640
1570    PRINT "  OK, THE COMPUTER WILL PLAY THE #'S."
1580    PRINT "WHO GOES FIRST  (1=COMPUTER  2=YOU)";
1590    INPUT F
1600    IF (F-1)*(F-2)=0 THEN 1630
1610       PRINT "PLEASE TYPE 1 OR 2 ";
1620       GOTO 1590
1630    LET T=3-F
1640 PRINT
1650 PRINT "HERE WE GO..."
1660 PRINT
1670
1680
1690 'MAIN MOVE SECTION
1700 LET T=3-T                          'T=1-2-1-2... OR, T=2-1-2-1...
1710 IF P=2 THEN 2000                   'SEE IF THERE ARE 2 PLAYERS
1720    IF T=2 THEN 2000                'OR IF IT'S THE &'S MOVE
1730       'GENERATE COMPUTER'S MOVE
1740       LET E=0                      'RESET MOVE QUALITY INDICATOR
1750       FOR J=1 TO 26                'CHECK POSSIBILITIES FOR EACH LETTER
1760          IF D(R(J),C(J))<65 THEN 1960  'IGNORE NON-LETTERS
1770             LET Q9=0               'RESET CHARACTER COUNTER
1780             FOR K=1 TO 33          'SEE HOW MANY MIGHT BE CAPTURED
1790                LET R1=R(J)+W(K)    'LOOK AT ALL CELLS WITHIN 1 CM
1800                IF (12-R1)*(R1-1)<0 THEN 1860  'BUT STAY ON THE BOARD
1810                   LET C1=C(J)+N(K)
1820                   IF (25-C1)*(C1-1)<0 THEN 1860
1830                      IF D(R1,C1)=32 THEN 1860  'IGNORE SPACES
1840                         IF D(R1,C1)=35 THEN 1860  'IGNORE #'S
1850                            LET Q9=Q9+1  'TALLY POSSIBLE CAPTIVES
1860             NEXT K
1870             IF Q9>E THEN 1930      'LOOK FOR BEST MOVE
1880                IF Q9=E THEN 1900   'AS GOOD AS BEST PREVIOUS?
1890                   GOTO 1960        'INFERIOR MOVE; TRY ANOTHER LETTER
1900                LET C9=C9+1         'TALLY POSSIBLE MOVES
1910                LET M(C9)=J         'STORE POSSIBLE MOVE
1920                GOTO 1960           'TRY NEXT LETTER
1930             LET C9=1               'RESET POSSIBLE MOVE COUNTER
1940             LET M(C9)=J            'STORE THE MOVE
1950             LET E=Q9               'UPDATE MOVE QUALITY INDICATOR
1960       NEXT J
1970       LET J1=M(1+INT(RND*C9))      'PICK AT RANDOM FROM BEST MOVES
1980       PRINT "THE #'S CAPTURE ";CHR$(D(R(J1),C(J1)))
1990       GOTO 2090                    'GO UPDATE DISPLAY, ETC.
2000 PRINT "THE ";P$(T);"'S CAPTURE";   'GET MOVE FROM HUMAN(S)
2010 LINPUT A$
2020 GOSUB 2730                         'PROCESS INPUT
2030 LET A$=SEG$(A$,1,3)                'WE ONLY NEED FIRST 3 LETTERS
2040 IF A$="STO" THEN 2350
2050    IF A$="RES" THEN 2330
2060       IF (90-A(1))*(A(1)-65)<0 THEN 2300  'LOOK FOR A LETTER
2070          FOR J1=1 TO 26            'LOOK FOR LEGAL MOVE
2080             IF A(1)<>D(R(J1),C(J1)) THEN 2290  'LOOK FOR THE LETTER
2090                FOR K=1 TO 33       'UPDATE BOARD
2100                   LET R1=R(J1)+W(K)  'CHECK CELLS WITHIN 1 CM
2110                   IF (12-R1)*(R1-1)<0 THEN 2160  'BUT STAY ON THE BOARD
2120                      LET C1=C(J1)+N(K)
2130                      IF (25-C1)*(C1-1)<0 THEN 2160
2140                         IF D(R1,C1)=32 THEN 2160  'IGNORE SPACES
2150                            LET D(R1,C1)=32+3*T  'ASCII FOR # OR &
2160                NEXT K
2170                LET M=M+1           'TALLY MOVES COMPLETED
2180                IF M=8 THEN 2270    'GAME OVER?
2190                   FOR K=1 TO 26
2200                      IF D(R(K),C(K))<65 THEN 2260  'ANY LETTERS LEFT?
2210                         'IF 2 PLAYERS, PRINT BOARD AFTER EVERY MOVE
2220                         'OTHERWISE, PRINT BOARD ONLY AFTER #'S MOVE
2230                         IF (P-2)*(T-1)<>0 THEN 2250
2240                            GOSUB 2890  'PRINT BOARD, THEN
2250                         GOTO 1700  'CONTINUE GAME...
2260                NEXT K
2270                GOSUB 2890          'PRINT FINAL BOARD, THEN
2280                GOTO 2390           'GO SEE WHO WON...
2290          NEXT J1
2300       PRINT "WHAT YOU TYPED IS NOT A LEGAL MOVE!"
2310       GOSUB 3130                   'PRINT LAST PART OF INSTRUCTIONS
2320       GOTO 2000                    'TRY AGAIN...
2330    PRINT "*** THE ";P$(T);"'S GIVE UP!  *** THE ";P$(3-T);"'S WIN!"
2340    GOTO 2560
2350 STOP
2360
2370
2380 'END-OF-GAME MESSAGES AND REPLAY QUERY
2390 FOR J=1 TO 12
2400    FOR K=1 TO 25
2410       IF D(J,K)=35 THEN 2460
2420          IF D(J,K)=35 THEN 2440
2430             GOTO 2470
2440          LET H2=H2+1               'TALLY &'S
2450          GOTO 2470
2460       LET H1=H1+1                  'TALLY #'S
2470    NEXT K
2480 NEXT J
2490 PRINT
2500 IF H1<>H2 THEN 2540
2510    PRINT "*** THIS GAME IS A TIE!"
2520    PRINT "EACH PLAYER HAS";H1;"CAPTIVES."
2530    GOTO 2560
2540 PRINT "*** THE ";P$((SGN(H2-H1)+3)/2);"'S WIN!!"
2550 PRINT "THE #'S HAVE";H1;"CAPTIVES AND THE &'S HAVE ";STR$(H2);"."
2560 PRINT
2570 PRINT "WANT TO PLAY AGAIN";
2580 LINPUT A$
2590 GOSUB 2730                         'PROCESS INPUT
2600 IF SEG$(A$,1,1)="Y" THEN 2620
2610    STOP
2620 PRINT "1=SAME SETUP  2=NEW SETUP ";
2630 INPUT A9
2640 IF (A9-1)*(A9-2)=0 THEN 2670
2650    PRINT "PLEASE TYPE 1 OR 2 ";
2660    GOTO 2630
2670 LET A$=""                          'AVOID REPEATING INSTRUCTIONS
2680 ON A9 GOTO 2690,1180
2690 MAT D=E                            'RESET DISPLAY
2700 GOTO 1440
2710
2720 'PROCESS INPUT; CHANGE LOWERCASE TO UPPER AND IGNORE ALL NON-LETTERS
2730 IF LEN(A$)>10 THEN 2860
2740    CHANGE A$ TO A
2750    LET A9=0
2760    FOR J=1 TO A(0)
2770       IF A(J)<96 THEN 2790         'CHECK FOR UPPERCASE
2780          LET A(J)=A(J)-32          'CHANGE LOWERCASE TO UPPER
2790       IF (90-A(J))*(A(J)-65)=>0 THEN 2810  'LOOK FOR LETTER
2800          GOTO 2830                 'ELSE IGNORE THIS CHARACTER
2810       LET A9=A9+1                  'TALLY ACCEPTABLE CHARACTERS
2820       LET A(A9)=A(J)               'STORE THIS CHARACTER
2830    NEXT J
2840    LET A(0)=A9
2850    CHANGE A TO A$
2860 RETURN
2870
2880 'PRINT THE BOARD
2890 PRINT
2900 FOR J=1 TO 12
2910    FOR K=25 TO 1 STEP -1           'IGNORE TRAILING SPACES
2920       IF D(J,K)=32 THEN 2970
2930          FOR L=1 TO K
2940             PRINT CHR$(D(J,L));
2950          NEXT L
2960          GOTO 2980
2970    NEXT K
2980    PRINT
2990 NEXT J
3000 PRINT
3010 RETURN
3020
3030 'INSTRUCTIONS
3040 PRINT
3050 PRINT "GOING IN TURN, THE PLAYERS (# AND &) CAPTURE ANY LETTER ON"
3060 PRINT "THE BOARD.  NOT COUNTING SPACES, ALL CHARACTERS WITHIN ONE"
3070 PRINT "CENTIMETER OF THE CAPTURED LETTER ARE ALSO CAPTURED AND"
3080 PRINT "CHANGE TO THAT PLAYER'S SYMBOL."
3090 PRINT
3100 PRINT "THE GAME ENDS AFTER FOUR TURNS OR IF ALL LETTERS HAVE BEEN"
3110 PRINT "CAPTURED.  THE PLAYER WITH THE MOST CAPTIVES WINS."
3120 PRINT
3130 PRINT "TO CAPTURE, TYPE ANY SINGLE LETTER SHOWN ON THE BOARD."
3140 PRINT "TYPE 'STOP' TO STOP OR 'RESIGN' TO RESIGN."
3150 PRINT
3160 RETURN
3170
3180 END
READY
```

HEATHKIT

Welcome to the home computer field, Heath Company! We're looking forward to your promised quality, service, and software support. And how about lower prices (see page 31) while you're at it?

Since 1926, when Ed Heath marketed his $199 Parasol airplane kit, Heath Company has been selling a wide variety of consumer electronic products, usually in kit form. Product categories have included Amateur Radio, hi-fi components, test instruments, auto and marine accessories, black and white and color TVs, as well as general consumer and educational products. And now Heath, a division of Scheumberger, Ltd., has announced its new line of home/personal computing products for hobby, home, educational and small business applications.

Many people are excited about the equipment, for the manufacturer has a half century reputation for quality, reliability, and service. Successful assembly of their kits frequently has been accomplished by persons with little or no electronic skills and knowledge. Instruction manuals have been thorough and accurate to meet the needs of such customers. Buying a Heathkit computer means you can expect that an accurate manual and optional educational materials will be available. Heath's attention to support, service and education are welcome additions to the computer field where some estimate 80-90% of kits never become operational.

Each of Heath's kits are trial-built by at least 100 people ranging from experts to first-time kit builders before they are presented to the public. Heath spokesmen claim that 'anyone' can build their kits. With Heath's reputation behind it, perhaps so. They've been busily training service representatives to deal with their computer line, and their reputation for service and troubleshooting support will doubtless be maintained in this area.

The company has made a strong committment to providing back up for its computer line in the form of software, complete documentation and service support, self-instructional programming courses, and a Heath Users' Group (HUG).

Their new line is built around two systems, one an 8-bit machine using the 8080A microprocessor, and the other incorporating Digital Equipment Corporation's (DEC) LSI-11, a 16-bit computer. System-compatible peripherals now include a TV-type terminal, a paper-tape reader/punch, a cassette-player/recorder, a 'hard copy' printing terminal, and serial and parallel interfaces. Input/output interfaces, additional memory, and supplementary software packages complete the initial product offerings, which will be available by September.

Heathkit H8 Digital Computer

THE H8

The H8 is built around a wired and tested CPU board containing an 8-bit 8080A microprocessor. It is designed for turnkey operation; a programmable speaker provides audio feedback as to whether you've performed an operation correctly (short beep) or not (long beep). Its intelligent front panel has a 9-digit 7-segment octal display and a 16-key octal keyboard.

The H8 features a built in 1K x 8 ROM (Read Only Memory) that contains a monitor program designed to permit you to load or store a program by pushing a single button. Register and memory contents can be dynamically displayed while a program is running.

Heath considered using the S-100 bus, but rejected it in favor of an in-house design. The 'Benton Harbor' or 'BH bus' as it will probably be known, uses 50-pin connectors. The motherboard has 10 slots. Interrupts are on-board; data and address locations are in easy-to-remember numeric sequences. The built-in connection power supply can handle up to 32 K of memory and two input/output interfaces. Bus specifications will be published soon.

COMPUTERS

Heathkit H11 Digital Computer

The H8's multi-tasking capability was demonstrated to those attending a June 1 press party at Heath Company headquarters in Benton Harbor, Michigan. As a game of 'Hangman' ran on a video display, the LED front panel display played a game of 'Chase' and the programmable speaker provided 'musical' accompaniment. Meanwhile the system monitor kept monitoring.

The H8 software package that comes with the system includes BH ('Benton Harbor') BASIC; an editor, TED-8; an assembler, HASL-8; a debugger, BUG-8 and the panel monitor, PAM-8. Extended BH BASIC is available at extra cost. System features include lower case output capabilities, command completion (e.g., you type just the first few letters of a command, then the system completes the command for you), tape handling and syntax error detection during input . BH BASIC, an adaptation to Dartmouth BASIC, includes PEEK, POKE, PIN, OUT, sin, cos, log and a user function to permit access to machine language routines. In BH BASIC all arguments are expressions; it runs in 8K. Extended BH BASIC, which runs in 12K, also includes strings and a number of other unique functions. Heath plans to make available source listings for the monitor and the input/output routines for BH BASIC's floating point package.

THE H11

The H11 is based on DEC's 16-bit LSI-11 processor which gives it the operating characteristics of a standard PDP-11 minicomputer; its instruction set is virtually identical to others PDP-11 computers. The system features a wired and tested CPU with 4Kx16 dynamic RAM, a compact switching power supply, a built-in cooling fan, a built-in monitor program and a 12-slot back panel providing room for 6 boards (each takes 2 slots).

Each H11 system comes with a complete DEC system software package containing an editor, PAL-11 assembler, linker, on-line debug package (ODT), input/output executive, BASIC and FOCAL. H11 purchasers may join DECUS (The Digital Equipment Computer Users Society), a clearinghouse for the more than 28,000 worldwide members who wish to exchange programs and information. The DECUS library contains about 800 programs which can run on the H11.

The H11 as presently configured will support up to 20K of memory; expansion is planned. The current H11 is paper tape oriented.

PERIPHERALS

Heath's product line includes interface and memory boards as listed in the 'Heathkit Prices' table. In addition, a number of other peripherals are available.

The H9 is an alphanumeric video terminal which will work with any digital computer. The system uses a 67 key ASCII upper case keyboard with an 80 character, 12 line format on a 12 inch CRT. Other features include cursor control, a batch mode, a plot mode, and a format option to display four 20 character columns of text. Baud rate is selectable from 110 to 9600. Standard serial interfaces include EIA, 20mA loop, and TTL input/output.

The H10 is Heath's paper tape reader/punch unit. It will function with any digital computer; standard one inch wide rolls or fanfold paper tapes are used. Tape is read at 50 characters per second (cps); the punch operates at 10 cps. The read and punch units may be operated simultaneously, H10 features include a copy mode for tape duplication, a built-in heavy duty power supply, and a stepper motor for reliable reader tape drive. The interface is standard parallel TTL.

Heath offers DEC's LA36 DEC Writer II as a 30 cps hard-copy device. Features include the ability to handle forms from 3 inches to 14 7/8 inches wide, 128 ASCII upper/lower case character set, half or full-duplex control and parity check

on output. The printing head is designed so that the last printed character is always visible. The 20 mA current loop interface is standard. A fanfold paper option and optional E1A interface are available.

A GE tape cassette player/recorder is offered by Heath as a mass storage device for their 8080-based H8.

A 6800-based trainer to teach machine language programming and interfacing will be available in October at a cost well below $200.

SUPPORT MATERIALS

Heath now has underway a number of self-instructional courses to accompany its computer line; several will appear as programmed instruction workbooks. A BASIC course is now almost complete and should be available in October at a cost of about $30; additional workbooks will be available at low cost. The 6800 trainer course to teach machine language programming and interfacing will also be available in mid autumn. Its approximately $90 price tag will cover the course plus the components discussed in the course.

Assembly language courses for the H8 and H11 should be arriving in late 1977 and early 1978, respectively. Heath has contracted with Dymax to write the H8 course, so some familiar folk are working on it: Don Inman (author of *People's Computers'* Data Handler series), Bob Albrecht (founder and former editor of this publication and author of its 'Tiny BASIC' series) and Jerry Brown (author of *Instant* BASIC). These courses will cost about $40-50 each, with additional workbooks available at lower cost.

HUG, the Heath Users' Group, will perform a number of educational functions, including exchange of information and programs. Heath plans to publish minor software revisions through HUG; major revisions will be available at a modest fee. Membership in HUG will be by subscription.

FUTURE PLANS

In the near future, Heath plans to develop applications software and produce educational courses, perhaps using computer-assisted instruction. We can look forward to seeing better graphics capability (including color), a floppy disk system for the H11, and a printer. Many ideas for future development are being considered, including plug in ROMs and a single-box system – *(e.g., CRT, CPU, floppies and keyboard in a single package.)*

Heath undoubtedly is moving in the right direction, with its emphasis on software, service and education. Future developments will depend in part on the wishes of Heath's customers; company spokesmen say Heath's way of doing business has long been characterized by an attitude of 'the customer will tell us what's wanted.' Do customers want to be able to buy completely assembled systems? FORTRAN? PASCAL? Applications software for small businesses? Anything is possible. Heath makes a welcome addition to home computer business.

Heathkit H10 Papertape Reader/Punch

Heathkit H9 CRT Terminal

HEATHKIT PRICES

Item	Description	Cost
H8	8080A based system includes a wired & tested CPU, documentation; software in audio cassette form.	$ 375
H8-1	4K Static RAM board.	$ 140
H8-2	A parallel I/O card with software.	
H8-3	A 4K chip expansion set.	$ 95
H8-5	A serial I/O interface board with 300 & 1200 band audio cassette interface.	$ 110
H9	Video terminal, upper case keyboard.	$ 530
H11	16-bit LSI CPU wired & tested; 4Kx16 dynamic RAM; built-in backplane; power supply with switching regulators 2 full circuit protection; DEC system software package.	$1295
H11-1	4Kx16 static RAM board.	$ 275
H11-2	Parallel interface.	$ 95
H11-5	Flexible serial interface	$ 95

Prices are not yet available for the GE cassette player/recorder and the LA36 DEC Writer II.

For additional information, write Heath Company, Dept. 350-26, Benton Harbor, MI 49022 for their *Computer Information Package.*

PERSONAL COMPUTER NETWORK

The PCNET (Personal Computer NETwork) Committee has been functioning in the Palo Alto area since the April Computer Faire. The committee's goal is the creation of regional (followed by national) personal computer networks for the computer-to-computer transfer of messages and files. A set of network protocols should be operable in 8K bytes of machine code, and are designed to be implemented in string BASIC.

The committee believes such a network should be attractive to personal computer users. Participation will be voluntary; you can decide to participate (or not) on any given day of network operation. Network functioning will be relatively insensitive to the absence of an appreciable fraction of member computers.

Our current thinking indicates the following tentative equipment required for participation in the network:

- A personal computer with 12 - 16K of RAM and string BASIC.
- An originate/answer MODEM capable of 300BPS

A message service – the ability to send a message (generally English text, although almost any file can be sent) is very valuable. It doesn't sound very dramatic, but it is surprising how powerful and efficiency-improving such a message exchange facility is. What keeps ordinary message services (telephone, telegraph, mail) from working as well seems to be a combination of factors: Too slow (mail); often hard to catch someone (phone); hard or time consuming to use (mail, telegrams); expensive in terms of characters per dollar (phone, telegrams); etc. A computer based message system overcomes most of these difficulties.

People regularly using such a system rapidly develop a whole new communication style. Most messages are brief – 500 characters or less. Content tends to be much more informal and direct than conventional media such as business letters.

Message system users move rapidly toward a computer-based personally oriented file system containing messages, distribution lists, text files, etc. The difference is one of kind, not just of degree.

The PCNET Committee is about to start a series of experiments. We welcome people with personal computer systems who would like to participate; we're especially interested in people in the Palo Alto, California dialing area. We are also most interested in similar network efforts elsewhere in the country. We'd like to avoid west coast chauvinism and want to work closely with people in other parts of the country.

Dave Caulkins
437 Mundel Way
Los Altos CA 94022
work: (415) 328-2411
home: (415) 948-5753

FORTRAN MAN

BY LEE SCHNEIDER & TODD VOROS

F-Man has returned from Junction City, where he has exchanged data with Ludwig Von Monitor in an effort to rid the Old Country of Transitoria of the deadly Count Algol. Doktor Debug and F-Man set a trap for their foe . . . and who should walk into the breakpoint but the Count's despicable accomplice Igor the File Snatcher! As Igor is unwilling to talk, desperate measures are required . . . a memory dump reveals shocking results! But then, even more terrible news! The good Doktor's beautiful daughter Parity fails to arrive with the next message at the local UART . . . Parity has been removed -- into the clutches of the evil and notorious Count Algol!

The arrival of such terrible news has a most randomizing effect on the good Doktor . . . .

But then suddenly, as if the mention of his name is a high-level signal, a sudden oscillation bursts out in the opcode room where Igor has been placed in HOLD mode . . .

Apparently unconcerned by Igor's escape, F-Man heads for the exit location . . .

Our Hero provides the correct answer, as usual . . . for just a few milliseconds later the crooked (but not too bright) Igor enters one of the many deserted segments of the Von Neuman works . . .

But then, just a short microsecond later . . .

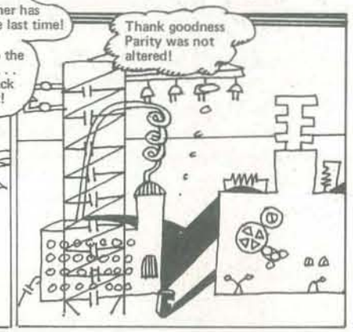
There is a moment of surprise . . . and then the Count, with speed rivaling that of F-Man himself, turns and branches away at high frequency! The noise generated is so great that some old control lines are triggered, and . . .

Responding to the emergency condition in less than a cycle, Fortran Man makes a long, unconditional jump from his level to the level below him . . .

. . . and not a nanosecond too soon!

Moments later, Parity is restored . . . and they discover that Igor had not been as fast, or as lucky, as they . . .

And before Parity can even wish him well, F-Man branches away once more and begins to rapidly trace the Count . . . passing through module after module filled with Stacks, Registers, Gates, Relays and Stored-Program Units . . . which in bygone days had kept the Von Neuman works operating . . . now deserted and crumbling into discrete componants . . .

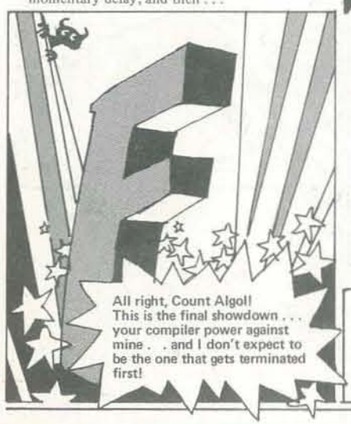


The chase continues, cycling through level after level . . . but even the great Fortran Man cannot operate in high-speed RUN mode for such extended periods without showing some signs of stress! He begins to tire . . . but then, as he cycles through an abandoned storage area . . .

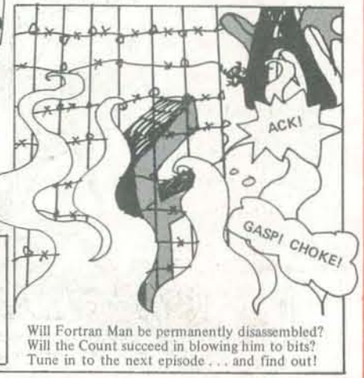
PAUSEing his memory cycle just long enough to refresh himself with an aged can of brewery's best, F-Man resumes the chase . . . at a much accelerated speed!

In close pursuit of the evil Count, F-Man branches out of the Beer Works . . . . . and, as the Transistorian sun begins to rise over the edge of the monolithic mountains, races towards the Count's dark and sinister castel . . .

The evil Count dissipates into the castle, with the fearless F-Man close behind . . . there is a momentary delay, and then . . .



Trapped by the falling edge of the gate output, Fortran Man is held fast . . . and as the evil Count Algol slowly descends the stairs to watch, small concealed output ports in the floor snap open . . . and begin to output streams of deadly Disassembly Gas!



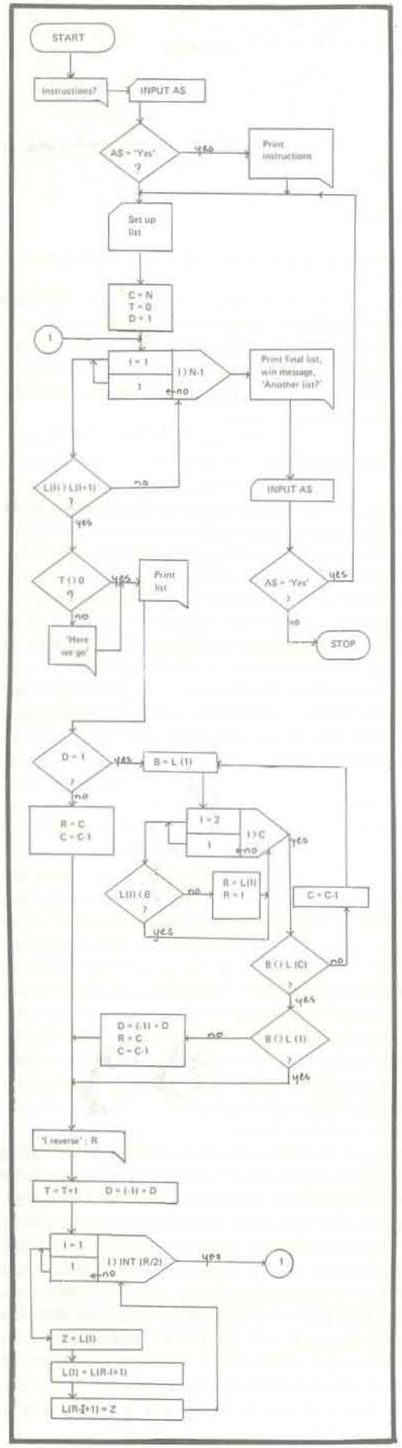
Will Fortran Man be permanently disassembled? Will the Count succeed in blowing him to bits? Tune in to the next episode . . . and find out!

INVERSE-REVERSE

BY CARL MAIN

Here is Carl Main's program to have the computer (instead of a human) play the game of REVERSE (see Vol 5, No 4). The object is to help kids discover an algorithm for playing REVERSE. The flow chart offered with the program listing is another way of helping explain how a program works — do you find it useful?

```
DO YOU WANT INSTRUCTIONS? !YES

I CAN DEMONSTRATE ONE ALGORITHM TO PLAY THE GAME
OF <<REVERSE>>. YOU WILL BE ASKED FOR THE NUMBER
OF NUMBERS IN YOUR LIST (FROM 1 TO 20 NUMBERS).
THEN YOU MAY GIVE ME ANY LIST OF NUMBERS OF THAT
LENGTH. I WILL USE THE ALGORITHM TO ARRANGE THE
LIST USING JUST THE ONE RULE FOR THE GAME OF
<<REVERSE>>. SEE IF YOU CAN DISCOVER MY ALGORITHM
AND THEN GO AND USE IT AS YOU PLAY <<REVERSE>>.

HOW MANY NUMBERS IN YOUR LIST? !4

IIM GAME ... TYPE THE LIST!
!3,2,4,1

HERE WE GO ... THE LIST IS:

3 2 4 1

I REVERSE 3 NUMBERS

4 2 3 1

I REVERSE 4 NUMBERS

1 3 2 4

I REVERSE 2 NUMBERS

3 1 2 4

I REVERSE 3 NUMBERS

2 1 3 4

I REVERSE 2 NUMBERS

1 2 3 4
THERE YOU GO! I DID IT IN 5 MOVES!!!!!

DO YOU HAVE ANOTHER LIST FOR ME TO PUT IN ORDER? !YES

HOW MANY NUMBERS IN YOUR LIST? !6

IIM GAME ... TYPE THE LIST!
!4.2,6.8,-3.7E-5,16.4,0,-1

HERE WE GO ... THE LIST IS:

4.2 6.8 -3.7E-05 16.4 0 -1

I REVERSE 4 NUMBERS

16.4 -3.7E-05 6.8 4.2 0 -1

I REVERSE 6 NUMBERS

-1 0 4.2 6.8 -3.7E-05 16.4

I REVERSE 4 NUMBERS

6.8 4.2 0 -1 -3.7E-05 16.4

I REVERSE 5 NUMBERS

-3.7E-05 -1 0 4.2 6.8 16.4

I REVERSE 2 NUMBERS

-1 -3.7E-05 0 4.2 6.8 16.4
THERE YOU GO! I DID IT IN 5 MOVES!!!!!

DO YOU HAVE ANOTHER LIST FOR ME TO PUT IN ORDER? !YES

HOW MANY NUMBERS IN YOUR LIST? !4

IIM GAME ... TYPE THE LIST!
!3,5,-1,6,5,10,3

HERE WE GO ... THE LIST IS:

3 5 -1 6 5 10 3

I REVERSE 6 NUMBERS

10 5 6 -1 5 3 3

I REVERSE 7 NUMBERS

3 3 5 -1 6 5 10

I REVERSE 5 NUMBERS

6 -1 5 3 3 5 10

I REVERSE 6 NUMBERS

5 3 3 5 -1 6 10

I REVERSE 5 NUMBERS

-1 5 3 3 5 6 10

I REVERSE 2 NUMBERS

5 -1 3 3 5 6 10

I REVERSE 4 NUMBERS

3 3 -1 5 5 6 10

I REVERSE 3 NUMBERS

-1 3 3 5 5 6 10
THERE YOU GO! I DID IT IN 8 MOVES!!!!!

DO YOU HAVE ANOTHER LIST FOR ME TO PUT IN ORDER? !NO
```


```
120  PRINT 'DO YOU WANT INSTRUCTIONS?':
130  INPUT A$
140  IF SUB(A$,1)='Y' THEN 960
150  DIM L(20)
160  PRINT
170  PRINT 'HOW MANY NUMBERS IN YOUR LIST?':
180  INPUT N
190  IF N<=0 THEN  STOP
200  IF ABS(N)=INT(N) THEN 240
210  PRINT
220  PRINT 'YOU MAY ONLY HAVE AN INTEGER NUMBER OF NUMBERS IN YOUR LIST
230  GOTO 170
240  IF N<=20 THEN 300
250  PRINT
260  PRINT 'RIGHT NOW I CAN ONLY MANAGE A LIST OF 20 OR FEWER NUMBERS.'
270  PRINT 'IF YOU WANT ME TO TAKE A LONGER LIST, PLEASE CHANGE THE DIM
280  PRINT 'STATEMENT IN LINE 150.'
290  GOTO 170
300  PRINT
310  PRINT 'IIM GAME ... TYPE THE LIST!'
320  MAT L=ZER(N)
330  MAT INPUT L
340  PRINT
350  REM- INITIALIZE THE COUNTER (T) AND INDICATORS (C & D)
360  T=0
370  C=N
380  D=1
390  REM- CHECK FOR WIN
400  FOR I=1 TO N-1
410  IF L(I)>L(I+1) THEN 560
420  NEXT I
430  REM- PRINT FINAL LIST L(1) TO L(N)
440  PRINT
450  FOR I=1 TO N
460  PRINT L(I);
470  NEXT I
480  PRINT
490  PRINT 'THERE YOU GO! I DID IT IN';T;'MOVES!!!!!'
500  PRINT
510  PRINT 'DO YOU HAVE ANOTHER LIST FOR ME TO PUT IN ORDER?':
520  INPUT A$
530  PRINT
540  IF SUB(A$,1)='Y' THEN 170
550  STOP
560  IF T<>0 THEN 600
570  PRINT 'HERE WE GO ... THE LIST IS:'
580  REM- PRINT LIST L(1) TO L(N)
590  PRINT
600  FOR I=1 TO N
610  PRINT L(I);
620  NEXT I
630  PRINT
640  PRINT
650  REM- DECIDE WHETHER WE ARE PUTTING NEXT LARGEST NUMBER TO FAR RIGH
660  REM- OR PUTTING NEXT LARGEST IN ITS FINAL RESTING PLACE.
670  IF D=1 THEN 710
680  R=C
690  C=C-1
700  GOTO 860
710  REM- SORT THROUGH THE UNORDERED PART OF THE LIST TO FIND THE
720  REM- LARGEST NUMBER LEFT TO BE PUT IN PLACE.
730  B=L(1)
740  FOR I=2 TO C
750  IF L(I)<B THEN 780
760  B=L(I)
770  R=I
780  NEXT I
790  IF B<>L(C) THEN 820
800  C=C-1
810  GOTO 730
820  IF B<>L(1) THEN 850
830  D=(-1)*D
840  GOTO 680
850  REM- REVERSE R NUMBERS
860  PRINT
870  PRINT 'I REVERSE';R;'NUMBERS'
880  T=T+1
890  D=(-1)*D
900  FOR I=1 TO INT(R/2)
910  Z=L(I)
920  L(I)=L(R-I+1)
930  L(R-I+1)=Z
940  NEXT I
950  GOTO 390
960  REM- PRINT THE INSTRUCTIONS
970  PRINT
980  PRINT 'I CAN DEMONSTRATE ONE ALGORITHM TO PLAY THE GAME'
990  PRINT 'OF <<REVERSE>>. YOU WILL BE ASKED FOR THE NUMBER'
1000 PRINT 'OF NUMBERS IN YOUR LIST (FROM 1 TO 20 NUMBERS).'
1010 PRINT 'THEN YOU MAY GIVE ME ANY LIST OF NUMBERS OF THAT'
1020 PRINT 'LENGTH. I WILL USE THE ALGORITHM TO ARRANGE THE'
1030 PRINT 'LIST USING JUST THE ONE RULE FOR THE GAME OF'
1040 PRINT '<<REVERSE>>. SEE IF YOU CAN DISCOVER MY ALGORITHM'
1050 PRINT 'AND THEN GO AND USE IT AS YOU PLAY <<REVERSE>>.'
1060 PRINT
1070 GOTO 150
1080 END
```

WRITING CAI

BY ELLEN NOLD

In 1973, when Ellen Nold was Director of the Undergraduate Writing Program at Stanford University, she began creating programs in PYLON, a variation of PILOT. These programs were successfully incorporated into the freshman classes in English composition and some freshman seminars. Ellen also wrote two programs for a course she was teaching on the poetry of e.e. cummings.

Ellen is now Director of the Communications Project at Stanford's School of Engineering, where she teaches engineers how to teach writing to other engineers. She reports there is a chance Exxon will provide the school with a grant to produce computer-assisted instruction (CAI) materials to teach technical writing.

For the many readers who've expressed an interest in more educational materials for home systems, here's a do-it-yourself opportunity. Ellen's suggestions provide an excellent template for creating your own CAI materials. Her recommendations speak to anyone wishing to use CAI, whatever the computer system or author language to be used.

The programs in the PILOT article (pages 11-15), written by Ellen Nold and Sallie Cannom, illustrate how to put Ellen's suggestions into practice. We'll publish more programs by Ellen and her associates in later issues of People's Computers.

RELATIONSHIP BETWEEN TEACHER AND COMPUTER

When writing a CAI Program, the computer is the medium for your 'voice', your teaching, the same way an audio tape is. When you say to a student through the computer medium 'I don't understand' or 'I think that must be right', and 'I' in the statement is *you,* not the computer. The computer has no voice except the one you (or other programmers) give it. Please don't slip into the fiction of 'Charlie the Computer' who is teaching a lesson. Please don't blame the limitations of understanding on mechanical difficulties in the computer ('I'm sorry I didn't understand—must have lost an electron'), the computer has enough of its own real mechanical problems.

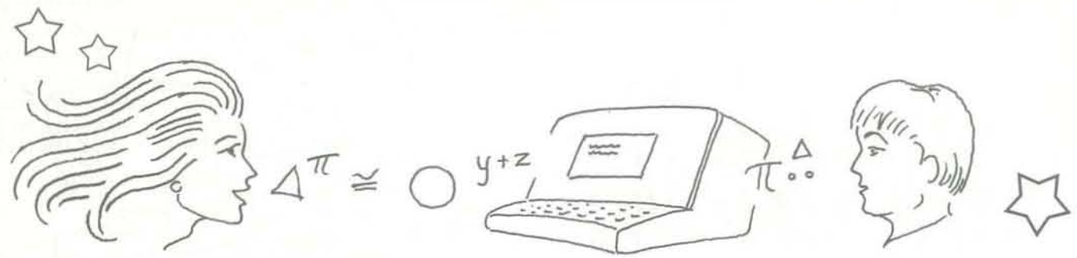
The computer has its limitations in responding to natural language, but try not to limit it more than necessary. Match for good spellings, common misspellings, first syllables, even strings of letters which may indicate a certain expected answer. Match for the negative answer to a yes/no question—they're much easier to catch. Use the computer like any other machine you appreciate—to the utmost of its capacities—but don't expect it to do everything. You don't use your washing machine to cook your food.

BEGINNINGS AND ENDINGS

The angle with which the student jumps into the computer lesson affects his/her response throughout. Realize that your student may be distracted, or even muddleheaded, when he/she sits down to do a lesson. The beginning communication ('HI. What's your name?') serves the same purpose between you speaking through the computer program and your student as it does between you speaking in person and the student: it says 'I see you there. Let's make contact and get on with the business of communicating.' Introducing the purpose of the program and asking if the student has done any prerequisite program provides an important mind-set. Use the student's name from time to time to focus his/her attention on the lesson.

Try not to end programs with a clunk. You may want to review the substance of the program; you may want to reflect some of the student's answers back to him/her; you may want to suggest another related (or unrelated) program; you may want to suggest further reading and/or consultation with a human source. If the program has done its best to raise the interest of the student, he/she will welcome these options. Please do not say at the end of the program: 'I'm tired now. Come back tomorrow.' Such statements may inhibit the student from trying this or another program again. When you teach through the computer, you never get tired. You might want to commiserate with the student after a particularly grueling lesson ('Bet you're exhausted by all these choices').

Dealing with incorrect answers is the main challenge in writing for the computer, and separates this kind of pedagogy from book writing and programmed text writing. In textbook writing, the teacher asks few questions and assumes assent and/or understanding from the student audience, perhaps repeating occasionally a point that he/she expects to be baffling or a little more difficult to grasp. In writing a programmed text, the teacher asks questions but immediately responds with the correct answer, whether or not the student has given it. In writing computer lessons, the teacher often matches for the correct answer and responds favorably if it is given.



CHOICE

It costs little more to give the student a choice, and choosing involves him/her much more intimately in the learning process. Typical choices are:

to go on with the program.
to try this example again
to try some more examples
to do Program B *without* having done Program A first.
to have the text for the next question typed
to review what he/she has learned so far
to be given an explanation of a rule
to have his/her answers saved for the teacher's editing

QUESTIONS, ANSWERS AND RESPONSES

The heart of teaching, and especially teaching through the computer, is question-asking. If you ask a hazy question, you'll be approximately three thousand times more likely to receive an answer that you cannot process. The first trick is to alert yourself to the difference between yes/no questions and others which require a non-dualistic answer. Do not ask *'Do you know* ways this generalization might be improved?' when you mean *'What are* the ways this generalization might be improved?' The second trick is to ask one question at a time. While the question 'What are Poe's three contributions to American literature, and how would you rate their importance?' elicits an answer fairly simply processed by the human mind, it is not so simple to program the computer to respond intelligently. The third trick is to first ask 'recall' questions and questions eliciting lower types of cognitive response and to then ask questions which involve interpretation and/or evaluation. In revising the question about Poe, we first want to see if the student can remember the contributions as we stated them in the lecture (or maybe as he/she sees them) and then whether he/she can weigh their relative importance. If we began with the importance question, we might get off the track because the student is missing vital information. The fourth and final trick is to give hints in the question as to the range of answers you expect so that you may intelligently respond to the student. For example, 'What is you feeling about statement 2?' is so open-ended that it precludes any response but a reflective one: 'So you feel XXX, do you?' If you would like to respond more systematically, you may ask, 'Do you feel confident about statement 2?' or 'What is your feeling about statement 2: good, bad or indifferent?'

If a wrong answer is given, the teachers should most likely explore it until the student either gives or understands the reasons for the correct answer. This exploration can take many forms: (1) the student may be asked to try again; (2) the student may be given a pointer or hint and asked to try again; (3) certain expected wrong answers can be matched and the student's evident misperceptions can be corrected; (4) the student may be asked to explain his answer in the hopes that in re-thinking and verbalizing the student may become aware of his/her misperception. In all cases, CAI writers should take care that they do not fall into the 'programmed learning' trap of responding to every (or most) answers with a bald statement of the correct answer, without explanation or critique.

SPACING AND PACING

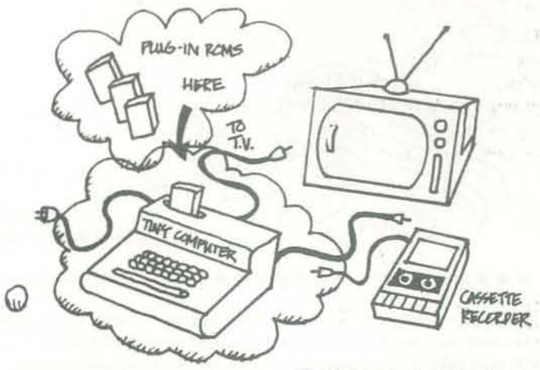
While writing and revising, you should be conscious of the pace of the lesson and its development. A lesson should not be composed solely of mind-blowing, rock-hard questions; neither should it be composed of quickly guessed, shallow questions. There should be breathing spaces for the student in which the computer types, but the student should not sit for minutes reading information typed to him. A rule of thumb is never to type more than five lines before the student is asked to respond. If there is a text the student will need, instead of typing it to him/her, remind him/her to bring the textbook or mimeographed copy. (You may decide to store the text in the memory and have it printed to the student if he/she requests it.)

When you change subjects or go to another example, double spacing is a good nonverbal cue. You may want to double space between an answer a student will have to spend a long time producing (such as a poem) and the computer's typed response: this kind of spacing preserves the delicate rhythm between the student and the computer lesson.

A computer lesson should never be used entirely for giving information; rather, it should be used primarily for allowing students to form and hone concepts of and attitudes towards the material in the course. Since the computer lesson is *interactive,* the student should *actively* practice the discipline taught in the course. Books provide information: computers provide practice.

MORE TINY BASIC

BY THE DRAGON



This concludes our brief introduction to Tiny BASIC. Here's a recap of the beginning of the series, in case you missed it back in Volume 5, Number 3, November-December, 1976.

Tiny BASIC AND THE CHEEP HOME COMPUTER

1975 was the year of the first build-it-from-a-kit personal computer – the MITS ALTAIR 8800. Others soon followed; today, two dozen companies offer personal computers . . . for the skilled hobbyist or computer expert.

Beginner, beware! Most of the 20,000 to 30,000 people who have acquired personal computers are highly skilled technologists: computer programmers, electronics engineers and technicians, experienced hobbyists, or people who have somehow acquired the high-technology skills necessary to assemble, trouble-shoot, repair, program, operate and understand their sophisticated cybernetic gadgets.

Now for the good news; your turn is coming!

1977 *could* be the year of the very inexpensive, off-the-shelf-ready-to-use personal computer for beginners. As easy to assemble as a hi-fi system: simply connect a couple of cables, plug in the Tiny Language of your choice, settle down with a good 'how-to' book, and learn to use your own computer. Or . . . plug in programs to play the games you have been reading about in our pages for almost 5 years (and more games are on the way!) Plug in a program to convert your color TV to a personal graphics machine, or music machine. Or a program to teach a child to read or do arithmetic or learn music.

Your ready-to-use personal computer might look like the one we will describe in this article and might cost less than $500, if some intrepid manufacturer would build and sell quite a lot of them. The same low cost computer could be a powerful tool for learning in elementary schools, or a home, or both. All we need are some tools: hardware, software, instructional materials, and lots of new ideas . . . the same system at home, or at school.

Here we go . . . *our* idea of a low-cost, ready-to-use, home/school/personal computer system. Our computer has four components.

1 The computer
2 A black and white or color TV
3 An inexpensive cassette tape recorder
4 One or more plug-in ROMs

Plug-in ROMs? A ROM is a Read Only Memory. Each ROM holds up to 4096 bytes* of pre-programmed memory . . . making it super easy for *beginners* to use the system. A ROM might contain:

- A programming language, such as BASIC or PILOT or PASCAL.
- A program to convert your TV into a fantastic light/sound show, with *you* at the controls.
- A program to play STAR TREK or Hunt the Wumpus or The Don Quixote Starship or . . .
- A program to convert your system into a powerful scientific, mathematical, statistical or financial calculator.
- A program to control household appliances, including the heating and lighting systems.
- ? ? ?

The key to this system is the ability to accept plug-in ROMs or something similar. Plug in one of more ROMs, hit a couple of keys, and GO! Change languages in seconds! A 4096 byte ROM will easily hold Tiny BASIC and a not-so-Tiny PILOT. Two such ROMs give you a more complete BASIC or other commonly used computer language. When better languages come along, get the ROMs and plug them in.

* A byte is a bunch of bits. A bit is a binary digit, 0 or 1. Usually a byte consists of 8 bits. In the near future, plug-in ROMs may hold up to 8192 bytes, or even 16384 bytes.

AND ON WE GO . . . .

Last time, you may recall we talked about the RND function, useful for generating 'random numbers' and the IF statement, which allows us to build in decision making capabilities in our programs. We finished the last article with a simple game-playing program called THE ESP MACHINE. Now, here is a modified version of THE ESP MACHINE.

```
100  PRINT "CAN YOU TELL ME WHAT I'M THINKING? LET'S FIND OUT."
110  PRINT "I WILL THINK OF A NUMBER FROM 1 TO 3 (1 OR 2 OR 3)."
120  PRINT "SO, YOU HAVE 1 CHANCE IN 3 OF GUESSING MY NUMBER."
130  PRINT "WHEN YOU WANT TO QUIT, TYPE -1 AS YOUR GUESS."
140  LET T=0
150  LET C=0

160  PRINT
170  PRINT "GUESS MY NUMBER . . . AND . . . GOOD LUCK!!!"
180  LET X = RND (1,3)
190  INPUT "WHAT IS YOUR GUESS? ";G
200  IF G = -1 THEN GOTO 250
210  LET T = T + 1
220  IF G=X THEN LET C=C+1:PRINT "THAT'S IT! MY NUMBER WAS ";X
230  IF G<>X THEN PRINT "AHA! FOOLED YOU. MY NUMBER WAS ";X
240  GOTO 160

250  PRINT
260  PRINT "YOU GOT ";C;" CORRECT IN ";T;" TRIES."
270  PRINT "THANKS FOR PLAYING! LET'S PLAY AGAIN SOMETIME."
999  END
RUN

CAN YOU TELL ME WHAT I'M THINKING? LET'S FIND OUT.
I WILL THINK OF A NUMBER FROM 1 TO 3 (1 OR 2 OR 3).
SO, YOU HAVE 1 CHANCE IN 3 OF GUESSING MY NUMBER.
WHEN YOU WANT TO QUIT, TYPE -1 AS YOUR GUESS.

GUESS MY NUMBER . . . AND . . . GOOD LUCK!!!
WHAT IS YOUR GUESS? 2
AHA! I FOOLED YOU. MY NUMBER WAS 1

GUESS MY NUMBER . . . AND . . . GOOD LUCK!!!
WHAT IS YOUR GUESS? 3
THAT'S IT! MY NUMBER WAS 3

GUESS MY NUMBER . . . AND . . . GOOD LUCK!!!
WHAT IS YOUR GUESS? 2
THAT'S IT! MY NUMBER WAS 2

GUESS MY NUMBER . . . AND . . . GOOD LUCK!!!
WHAT IS YOUR GUESS? 1
AHA! I FOOLED YOU. MY NUMBER WAS 3

GUESS MY NUMBER . . . AND . . . GOOD LUCK!!!
WHAT IS YOUR GUESS? -1

YOU GOT 2 CORRECT IN 4 TRIES.
THANKS FOR PLAYING! LET'S PLAY AGAIN SOMETIME.
```

■ ← Cursor. What next, sir or madam?

Our program has two slight improvements over the program in the previous issue (Volume 5, Number 6, May - June, 1977, page 35).

(1) It keeps track of how many times the player tries to guess the computer's number and how many correct guesses she or he made.
(2) It gives the player a method for quitting the game. To quit, you type -1 as your guess. When you quit, the computer then tells you how many you got correct out of how many tries.

The number of tries is called T. The variable T appears in Lines 140, 210 and 260. In Line 140, prior to any guesses, T is set to zero. Then, in Line 210, after each guess, T is increased by one. Finally, in Line 260, the value of T is printed for all the world to see.

The number of correct guesses is called C. The variable C appears in Lines 150, 220, and 260. In Line 150, prior to any guesses, C is set to zero. Then, in Line 220, if the guess (G) is a correct guess, C is increased by one (LET C = C + 1) and a message is printed to let the player know she or he has made a correct guess. We will describe Line 220 in more detail presently. Meanwhile, in Line 260, the value of C is printed in the 'end of game' message.

We will explain Line 220. This line has two statements, separated by a colon.

```
220  IF G=X THEN LET C=C+1:PRINT "THAT'S IT! MY NUMBER WAS";X
```

First statement — Colon — Second statement

When the guess is correct, G = X will be TRUE. In this case the computer will LET C = C + 1 and then will do PRINT "THAT'S IT! MY NUMBER WAS";X.

But, if the guess is *not* correct, G = X will be FALSE. In this event, the computer will *not* do the rest of the line. That is, the computer will *not* do anything to the right of the word 'THEN'. Instead, it will go immediately to Line 230.

Here is another way to look at it.

```
220  IF G=X THEN LET C=C+1:PRINT "THAT'S IT! MY NUMBER WAS";X
```

- If G=X is TRUE, do all this stuff.
- If G=X is FALSE, *don't* do any of this.

We could have (but didn't) combined Line 140 and 150 into single line, as follows.

```
140  LET T=0 : LET C=0
```

1st statement — Colon — 2nd statement

Using more than one statement per line saves space. Programs are shorter and use up less memory space inside the computer. However, be careful: don't sacrifice readability just to save space!

OUR NEXT ADVENTURE

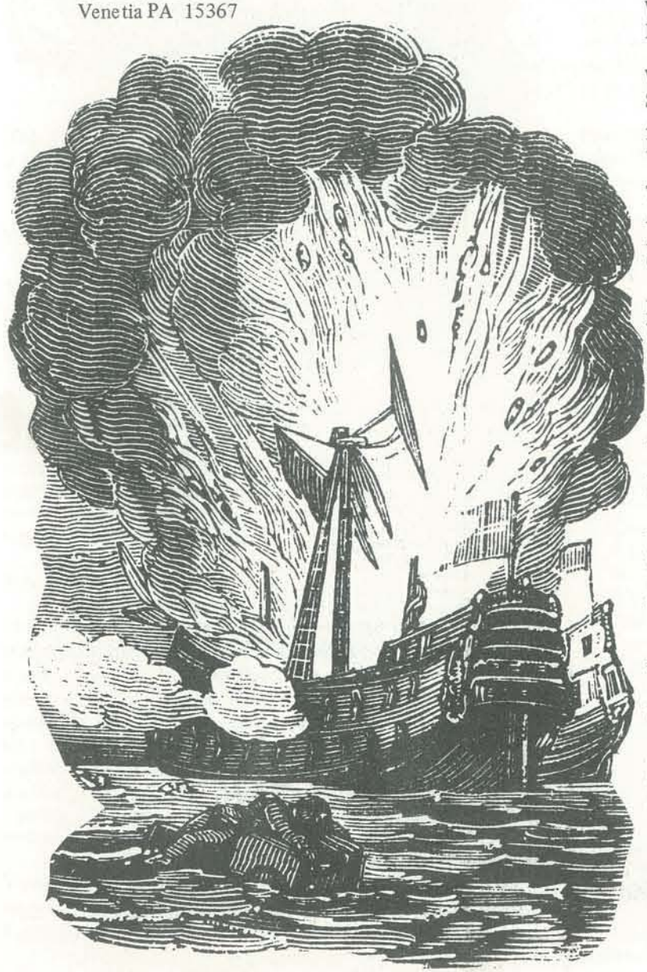
The people who brought you Tiny BASIC and Tiny PILOT are at it again: see our September - October issue for the beginning of the next Tiny Quest!

The Wreck of the Sequential Files

(To be sung to the tune of *The Wreck of the Edmund Fitzgerald)*

I'm writing to tell you that I like the magazine's new (Fortran?) format (statement?). Here's a modified version of *"The Wreck of the Edmund Fitzgerald"*. I would like to exchange games with anyone who wants to; I have lots of Streks–mostly in BASIC and a few unfinished.

Doug - Dit-Dit - Philips
McCombs Rd.
RD No. 2, Box 329
Venetia PA 15367



The legend lives on from the C.P.U. down
To the big loop they call real-time shuffle.
The loop it is said, never gives up its dead
When the skies of November turn gloomy.

With a load of programs eighty thousand lines more
Than the Sequential Files take up empty.
That good if and when was a bone to be chewed
When the gales of November came early.

The wind in the wires made a tattle tale sign
As a card broke over the reader,
And every file knew, as the system did too,
'Twas the witch of November come stealin'.

The dawn came late, swappin' had to wait
When the gales of November came slashing.
When afternoon came it was workin' strain
In the face of the Interface breakin'.

When data check came the machine came on deck
Sayin 'fellas, its too rough to check ya'.
At seven p.m. a main circuit fused in;
He said 'fellas, it's been good to know ya.

The files wired in, they had B.S. comin' in
And the good flip and flop were in peril,
And later that night, its display out of sight,
Came the crash of the Sequential Files.

Does any byte know where the real time clock goes
When the lists run printers to blotters?
The scanners all say they'd made hard copy bay
If they'd put fifteen more micros behind her.

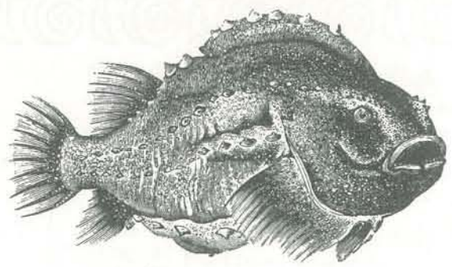
They may have split up, they may have been scratched,
They may have broke up and took garbage.
And all that remains are the records and games
Of the files and the runs and the listings.

Loop Huron rolls, Synthesizer sings
In the lines of her random access bytes.
Old Michigan streams like a young job's dreams;
The module and kits are for sportsman.

And further on down the disk data files take in
Whatever the teletype sends
And the processors go, as the monitors know,
With the files of November remembered.

In a musty old core in Detroit they prayed
In the teletypes interface link up.
The processor clock chimed twenty-nine times
For each line in the Sequential Files.

The legend lives on from the C.P.U. down
To the big loop they call real-time shuffle.
Synthesizer they said, never gives up her dead
When the gales of November come early.



8080 MATRIX SUBROUTINES

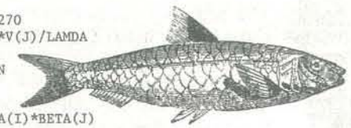
BY TIM SCULLY

Way back when, Tim Scully founded Aquarius Electronics and began designing, building, and marketing biofeedback equipment. Most recently, he has been applying biofeedback instruments, systems, and techniques for use in drug rehabilitation programs. One approach has involved using an 8080A microcomputer system for computer analysis of EEG, in an effort to find EEG correlations of psychological events. Such work frequently requires use of multivariate statistical analysis, which involves matrix algebra. Tim has sent People's Computers *two subroutines which we hope will be of use to others doing such analysis. They are written in Altair Extended BASIC, modified to run on an 8080 system with 64K RAM.*

MATRIX INVERSION ROUTINE

This subroutine inverts the square matrix E(ND,ND). Note that the calling program will normally have DIMensioned this matrix and hence it is not reDIMensioned in this subroutine.

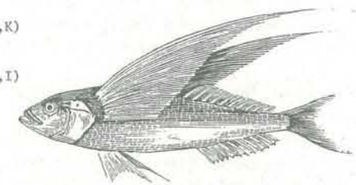
```
60000 DIM C(ND,ND):DIM ALPHA(ND):DIM BETA(ND):DIM U(ND):DIM V(ND)
60010 FOR I=0 TO ND
60020 E(I,I)=E(I,I)-1
60030 C(I,I)=1
60040 NEXT I
60050 N=0
60060 FOR I=0 TO ND
60065 IF E(N,N)=0 THEN 60080
60070 ALPHA(I)=E(I,N)/E(N,N)
60080 BETA(I)=E(N,I)
60090 NEXT I
60100 I=0
60110 K=N:S1=0:S2=0
60120 S1=S1+C(I,K)*E(K,N)
60130 S2=S2+C(K,I)*E(N,K)
60140 IF K=ND THEN 60160
60150 K=K+1:GOTO 60120
60160 U(I)=S1:V(I)=S2
60170 IF I=ND THEN 60190
60180 I=I+1:GOTO 60110
60190 J=N:S1=0
60200 S1=S1+V(J)*E(J,N)
60210 IF J=ND THEN 60230
60220 J=J+1:GOTO 60200
60230 LAMDA=S1+E(N,N)
60240 FOR I=0 TO ND
60250 FOR J=0 TO ND
60255 IF LAMDA=0 THEN 60270
60260 C(I,J)=C(I,J)-U(I)*V(J)/LAMDA
60270 NEXT J,I
60280 IF N=ND THEN RETURN
60290 FOR I=N+1 TO ND
60300 FOR J=N+1 TO ND
60310 E(I,J)=E(I,J)-ALPHA(I)*BETA(J)
60320 NEXT J,I
60330 N=N+1:GOTO 60060
```



MATRIX ROOT SUBROUTINE

This subroutine finds the roots (eigenvalues) of the symmetric matrix A(ND,ND) and returns them as the diagonal elements of A(ND,ND); the original matrix is destroyed in the process. The eigenvectors of the matrix A(ND,ND) are returned in the matrix S(ND,ND). The subroutine is iterative. If reduced accuracy is acceptable, line 61090 may be changed to increase the maximum allowable value of off-diagonal elements (presently set at 1×10^{-6}). This will speed execution of the program.

```
61000 'S(ND,ND) AND B(ND,ND) MUST BE DIMENSIONED BEFORE ENTRY
61001 'THIS ROUTINE FINDS THE EIGENVALUES AND
61002 'EIGENVECTORS OF THE SYMMETRIC MATRIX A(ND,ND)
61003 'BY THE METHOD OF JACOBI
61010 IC=0:V=0
61020 FOR I=0 TO ND
61030 FOR K=0 TO ND
61040 V=V+A(I,K)*A(I,K)
61050 NEXT K,I
61060 FOR I=0 TO ND
61070 V=V-A(I,I)*A(I,I)
61075 S(I,I)=1
61080 NEXT I
61085 V=SQR(V)
61090 VF=(1E-06)*V
61100 V=V/(ND+1)
61110 Q=1
61120 P=0
61130 IF ABS(A(P,Q))<V THEN 61360
61140 IC=1
61150 L=-A(P,Q)
61160 U=(A(P,P)-A(Q,Q))/2
61162 IF U=0 THEN SG=1 ELSE SG=SGN(U)
61163 IF L=0 THEN W=0:GOTO 61180
61170 W=SG*L/SQR(L*L+U*U)
61180 ST=W/SQR(2*(1+SQR(1-W*W)))
61190 CT=SQR(1-ST*ST)
61200 FOR I=0 TO ND
61210 B(I,P)=A(I,P)*CT-A(I,Q)*ST
61220 B(I,Q)=A(I,P)*ST+A(I,Q)*CT
61230 IP=S(I,P)*CT-S(I,Q)*ST
61240 IQ=S(I,P)*ST+S(I,Q)*CT
61242 S(I,P)=IP
61244 S(I,Q)=IQ
61250 NEXT I
61270 B(P,P)=A(P,P)*CT*CT+A(Q,Q)*ST*ST-2*A(P,Q)*ST*CT
61280 B(Q,Q)=A(P,P)*ST*ST+A(Q,Q)*CT*CT+2*A(P,Q)*ST*CT
61290 B(P,Q)=(A(P,P)-A(Q,Q))*ST*CT+A(P,Q)*(CT*CT-ST*ST)
61292 A(P,P)=B(P,P):A(Q,Q)=B(Q,Q):A(P,Q)=B(P,Q)
61300 A(Q,P)=A(P,Q)
61310 FOR I=0 TO ND
61315 A(I,P)=B(I,P):A(I,Q)=B(I,Q)
61316 NEXT I
61317 FOR I=0 TO ND
61320 A(P,I)=A(I,P)
61330 A(Q,I)=A(I,Q)
61340 NEXT I
61360 IF P=Q-1 THEN 61380
61370 P=P+1:GOTO 61130
61380 IF Q=ND THEN 61400
61390 Q=Q+1:GOTO 61120
61400 IF IC=1 THEN 61430
61410 IF V<=VF THEN RETURN
61420 GOTO 61100
61430 IC=0
61440 GOTO 61110
```



THE DATA HANDLER USERS MANUAL:

PART 4

BY DON INMAN

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 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.

The first semester course consists of nine two-hour class sessions, the first two of which were spent constructing the systems. To recap our series, Part 1 (Vol 5, No 4) covered computer specification, computer notation and use of the keyboard. Part 2 (Vol 5, No 5) covered use of registers, the instructions LDA, STA, and JMP, and the use of a simple data transfer program. Part 3 (Vol 5, No 6) covered the addition instructions ADC, CLD, and CLC, use of immediate mode addressing, and use of absolute addressing. This article, Part 4 in our series, covers the contents of the sixth class session: indexed addressing.

A summary of instructions taught through session VI is on the facing page; a description of the Data Handler is on page 46.

SESSION VI - INDEXED ADDRESSING

If we are to perform arithmetic problems (such as addition), with more than two numbers, the IMMEDIATE and the ABSOLUTE modes become very cumbersome to use. Our programs become very long and are too specific. The more general a program is, the easier it is to adapt to new data. In order to make our programs more general, we will introduce the concept of LOOPS and INDEXED ADDRESSING.

MNEMONIC CODE FOR DATA TRANSFER PROGRAM

BYTES	LABEL	INSTRUCTION	OPERAND	COMMENTS
2		LDX	0	Load index with zero (register X).
3	LOOP	LDA	FIELD1, X	
3		STA	FIELD2, X	
1		INX		Increment count (register X).
2		CPX	5	Compare for end of loop.
2		BNE	LOOP	

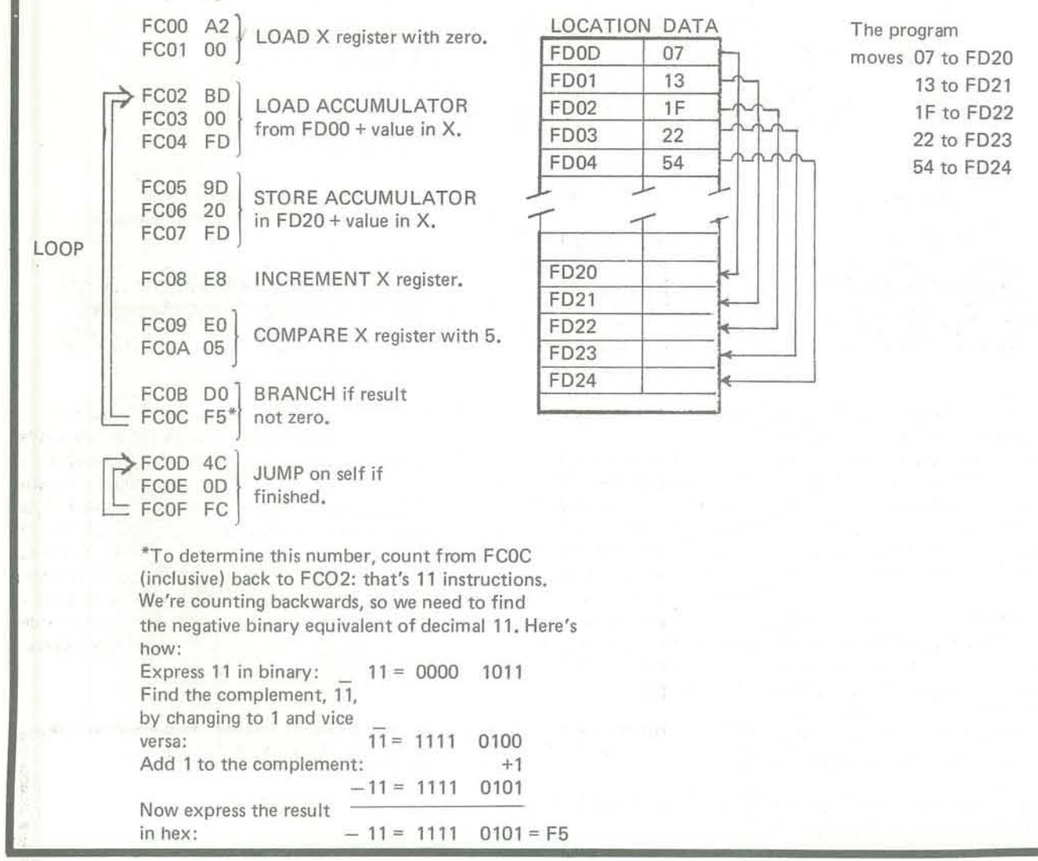
Our first program will be a simple transfer of data in memory using these new techniques. We will use as reference Example 6.4 on page 76 of the *MOS 6500 Microcomputer Family Programming Manual*, Jan 1976, second edition. New instructions for you are LDX, INX, and CPX. The INDEXED ADDRESSING mode is also new.

The X register will be used to control the program through the loop five times. We set it to zero initially and count each time we pass through the loop. When we reach five, we will exit from the loop. The number 5 was chosen since we will move 5 pieces of data. We have incremented our counter X in the following example, but we could just as well have written our loop to count downward as in Example 6.5 on page 77. We would then decrement the X register.

FIELD1 refers to the original locations in which you've chosen to store data; for this program, we'll let FIELD1 start at location FD00. As X is incremented, data is loaded from FD00, FD01, FD02, FD03, and FD04.

FIELD2 refers to the final locations chosen for this data transfer program. Let's let FIELD2 start at location FD20 in this program. As X is incremented, the data is stored in FD20, FD21, FD22, FD23, and FD24.

DIAGRAM OF DATA TRANSFER PROGRAM



*To determine this number, count from FC0C (inclusive) back to FC02: that's 11 instructions. We're counting backwards, so we need to find the negative binary equivalent of decimal 11. Here's how:

Express 11 in binary: 11 = 0000 1011

Find the complement, $\overline{11}$, by changing to 1 and vice versa: $\overline{11}$ = 1111 0100

Add 1 to the complement: +1

−11 = 1111 0101

Now express the result in hex: −11 = 1111 0101 = F5

First, LOAD the program by the procedure previously discussed.

Second, LOAD the data shown in the diagram into locations FD00 through FD04.

Third, LOAD the INITIALIZE VECTORS at FFFC and FFFD.

Fourth, EXAMINE all locations and CORRECT any errors.

Fifth, RUN the program.

Last, EXAMINE locations FD20 through FD24 to see if the program ran successfully.

Now that you were successful let's try an addition program to add the five numbers together. A few modifications are necessary.

SUMMARY OF INSTRUCTIONS TAUGHT THROUGH SESSION VI

MNEMONIC	INSTRUCTION
ADC	7D
BNE	D0
CLC	18
CLD	D8
CPX	E0
DEX	CA
INX	E8
JMP	4C
LDA	A9 (immediate mode)
	AD (absolute mode)
	BD (absolute,X)
LDX	A2
STA	8D (absolute mode)
	9D (absolute, X)

ADDITION OF FIVE NUMBERS

LABEL	ADDRESS	INSTR/DATA	MNEMONIC	COMMENTS
	FC00	A2	LDX	Load index register
	FC01	00		with zero.
	FC02	D8	CLD	Clear decimal mode.
	FC03	18	CLC	Clear carry.
	FC04	BD	LDA	Load accumulator (Absolute, X)
	FC05	00		from location FD00 + value in X register.
	FC06	FD		
LOOP1	FC07	7D	ADC	Add to accumulator with carry
	FC08	01		from location FD01 + value in X register.
	FC09	FD		
	FC0A	8D	STA	Store accumulator (Absolute)
	FC0B	50		into location FD50.
	FC0C	FD		
	FC0D	E8	INX	Increment the X register.
	FC0E	E0	CPX	Compare X
	FC0F	04		with 4.
	FC10	D0	BNE	Branch if not equal to zero
	FC11	F5		back to LOOP1.
LOOP2	FC12	4C	JMP	Jump if done
	FC13	12		to self LOOP2.
	FC14	FC		

Data to be loaded: 07 in FD00, 13 in FD01, 1F in FD02, 22 in FD03, and 54 in FD04
Initialize vectors: 00 in FFFC, FC in FFFD.

The answer will appear in FD50.

You will notice we have added a new column in our program on the left. Labels are used at key points so that various branch points and loop origins can be more easily located. Comments should contain information telling the reader which label is to be used.

It should also be noted that our counter only goes to four although we are adding five numbers. This results due to the fact that we load the first number and then add to that FOUR additional numbers. The partial sum is stored in location FD50 and is repeatedly added to each new number.

The number of addends can be readily changed by providing the necessary data and changing the value in location FC0F from 04 to one less than the number of addends.

TRACE OF DATA TRANSFER PROGRAM

PROGRAM STEPS	ACCUMULATOR	MEMORY FD50	X REG.	COMMENTS
INITIAL	??	??	??	Contents unknown
FC00 - 01	??		00	LDX
FC04 - 06	07			LDA from FD00
FC07 - 09	07 + 13 = 1A			ADC from FD01
FC0A - 0C		1A		STA FD50
FC0D			01	INX
FC0E - 0F				04 - 01 ≠ 0
FC10 - 11				Branch back
FC07 - 09	1A + 1F = 39			ADC from FD01 + 1
FC0A - 0C		39		STA FD50
FC0D			02	INX
FC0E - 0F				04 - 02 ≠ 0
FC10 - 11				Branch back
FC07 - 09	39 + 22 = 5B			ADC from FD01 + 2
FC0A - 0C		5B		STA FD50
FC0D			03	INX
FC0E - 0F				04 - 03 ≠ 0
FC10 - 11				Branch back
FC07 - 09	5B + 54 = AF			ADC from FD01 + 3
FC0A - 0C		AF		STA FD50
FC0D			04	INX
FC0E - 0F				04 - 04 = 0
FC10 - 11				No branch!
FC12 - 14				Loops on self DONE

ANSWER= AF

ORIGINAL STORE
FD00 07
FD01 13
FD02 1F
FD03 22
FD04 54
FFFC 00
FFFD FC

A TRACE is a chart that shows temporary results as a program runs. Traces are especially useful in finding bugs in programs. To create a trace, select the registers and/or memory locations of most interest to you, then step-by-step go through the program 'playing computer' and writing down the contents of each high-interest location at each step.

This is a trace of the preceding program. We chose to look at the accumulator, memory location FD50 (where our answer should appear) and the X register.

ADDITION PROGRAM

LABEL	ADDRESS	INST/DATA	MNEMONIC	COMMENTS
START	FC00	A2	LDX	Load X register
	FC01	04		with 4.
	FC02	D8	CLD	Clear decimal mode.
	FC03	18	CLC	Clear carry.
	FC04	BD	LDA	Load accumulator (Abs, X)
	FC05	00		from location FD00 + value in X register.
	FC06	FD		
LOOP1	FC07	7D	ADC	Add to accumulator with carry
	FC08	01		from location FD01 + value in X register.
	FC09	FD		
	FC0A	8D	STA	Store accumulator (Abs)
	FC0B	50		into location FD50.
	FC0C	FD		
	FC0D	CA	DEX	Decrement the X register.
	FC0E	D0	BNE	Branch if not equal to zero
	FC0F	FA		back to LOOP1.
LOOP2	FC10	4C	JMP	Jump to
	FC11	10		LOOP2.
	FC12	FC		

This program performs the same function as the preceding program, to add 5 numbers. The only difference is the use of the X register to count down from four rather than counting up. Decrementing the X register saves two programming steps (CPX) which were necessary in the preceding program.

For practice, write programs using as guidelines the programs in this article and the *Programming Manual* (eg. Example 6.5, page 77). A complete discussion of index registers and index addressing concepts is contained in chapter 6 of the *Programming Manual*.

The next in this series of articles will cover flow charts, double precision addition via flow charts, 8 bit multiply using a trace, and the instructions ASL, BCS, and DEX.

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.

MORE ON WOMEN AND COMPUTERS

FROM JEF RASKIN

Dear Ms Ran,

Since I was the author of the article in Dr Dobb's that helped you decide to get a Polly 88 computer, I thought that I'd be an appropriate person to try to answer some of your questions. Besides, I feel responsible (but neither guilty nor negligent).

First of all: you need know *no* mathematics in order to program. For years I taught a programming course (at UC San Diego) to hundreds of people who, like yourself, find even high school algebra a closed book – or at least an uninviting one. The idea that computers are basically mathematical creatures is false, but promulgated by the fact that most programmers are mathematically adept. When I taught I used a language, FLOW, that I invented that is so amathematical that it can't even add or subract. Thus I proved to my satisfaction (and to my students' satisfaction) that *anybody* can learn to program. The only prerequisite is 6th grade literacy. From your letter I can see that you have more than the minimum requirements to learn to program.

You do not need to know hexadecimal, octal or binary arithmetic to program. You do not need to know the secret inner workings of the computer. You do not need the arcane jargon of the computer scientist. You can learn to program. The question that you legitimately ask is how and where.

I know that the manuals that come with *all* the brands of microcomputers are inadequate for the newcomer. Believe it or not, Poly's are among the better ones! You should see some of the others. Most of the books in the field are also hopeless. As an author in the field I can see some of the reasons why. A book for the complete beginner has to be pegged to some *particular* computer and a *particular* version of some language. It can't be written for all or even most computers. They vary in detail too much. The beginner is just as floored by a picayune detail as by a general concept. In fact it takes a while before you can see which is which. In any case it has not been judged economically feasible by any company to come out with a text for real beginners (as you say, not dumb people, just those without a computer or scientific or mathematical background). I'd love to write one. I know how. You might ask: How much does it cost to write such a book? It might take me six months full time, plus I'd have to have an editor part of the time. The book then has to be TESTED. Every chapter and paragraph has to be tried on a number of people to see that it really works. I am sure you'd love to be one of the testers. It might cost, just in my time alone, some $15,000. The editor has to be paid, then there are printing and distribution costs. An investment of $30,000 would be an absolute minimum. Since computers are selling so well without such a book it is hard to get a manufacturer to go out of its way to make such an expenditure. Maybe your letter and this reply will get the ball rolling.

Another thing that you mention is the sexist nature of the computer world. This is a valid observation, there are very few women in the microcomputer (or other computer) field. In my classes there were nearly as many women as men. A glance at my course records shows that, given a computer course that did not assume the typical male point of view (you all know what a transistor is . . .), women do as well as men in programming. This is, of course, what I expected to find if I taught well. I have taught other subjects besides computer programming and know that there is no difference between women, third world people, and the 'normal' college population in learning ability.

If you have particular questions on the Poly-88 I'd be glad to try to answer them. If you wonder why I didn't mention typing expenses in my comments on writing books it is because my Poly-88 does my typing for me using a text editor that I wrote. Another random comment that should make you feel better is that even though I am a professional computer scientist, I had to call Polymorphic Systems a number of times to get explanations on stuff that the manuals supposedly covered.

I agree with most of the responses that were printed to your letter but I feel that Mr Inman was right in saying that your problem was that you are a beginner – but he was wrong in not recognizing just how much more of a beginner you are because you are female. This culture *does* make it harder for females to acquire the mental skills that make doing anything technological easier. Ms Owicki's letter made some good points but she labors under the idea that you have to be familiar with math to program.

I make the same comment to Ms Ahlgren, and ask her if CROMEMCO would want to sponsor a real beginner's book? I think Ms Liff hits the nail on the head with her reply in regard to the difficulties of females in technical fields.

One positive idea: if People's Computer Company, or anyone, would like to set up a course (which won't help you, living in Michigan, if it happens here) and handle the administration of it, I would offer to teach it with the guarantee that any person who can read and write who takes the course will learn how to program. Such a course does not belong in a university but among the people.

Jef Raskin
Box 511
Brisbane CA 94005

FROM ANNETTE RAN

Dear *People's Computers*,

I was really pleased with all the thoughtful responses to my letter. As Mr Raskin has noted, Mr Inman is both right and wrong in his contention that my problem is that I am a novice. A female novice comes to computing with more deficit in her background that do males. However, in all fairness to Mr Inman I should point out that it is more acceptable in our culture for a woman to admit that some element of technology is incomprehensible, but there are many men who face the same problem but are ashamed to admit it. One of the positives of the women's movement is that the issues it confronts are not just women's problems but human problems and their solution benefit not only women but all of us.

I think that the issue of sexism in science was detailed thoroughly in Professor Liff's letter and that the suggestions she and the others make have a great deal of merit. I hope that someone picks up on them.

As was pointed out, some of the problems I am having are because I am 1) primarily interested in software and 2) dealing with an infant industry which has hardly begun the process of debugging itself. It is to those two interrelated problems that I would like to address myself in this round of our dialogue.

When I first got my Poly, I rubbed my hands with glee and sat down to try some of the neat games that were in various books I had bought. It took two days, three people and numerous long distance calls to Ann Arbor, Michigan where we made our purchase, to figure out how to load our BASIC cassette tape.

Finally with that done, I sat down once again to explore the wide world of computing. Imagine my surprise when I learned that there was no such thing as plain BASIC, that every computer company had its own variation and that none of the programs in any of those books and magazines worked in my Poly. Nobody had bothered to tell me that fact and I was flabbergasted. Talk about anarchy! So instead of using programs, we all spent hours trying to figure out how to debug programs. Net result – more long distance calls – a few to California where Poly is located (nice people but not helpful to us novices) – and not a single program that works well. So my number one suggestion, plea, etc. is that the computer manufacturers hold a summit conference and develop a mutually compatible BASIC. Since I am basically a pessimist and don't believe such a conference will ever be held, my next suggestion is that someone publish a chart of the various BASICs showing how to translate one into the other. That probably is unrealistic too.

I have a feeling that many people probably enjoy the challenge of perfecting their BASIC and making it work but I am not a tinkerer and want to be able to type in an already developed program and have it work the first time. Another thing which occurs to me is that many manufacturers rely on their customers to complete the job of debugging their hardware and software. Once again that favors the male and his skills and leaves females like me out in the cold.

If the personal computer industry is to expand, it must appeal to a wider market than it does currently. Small businessmen, researchers, teachers all are potentail users of computers but they want equipment which does not require that they learn a whole new discipline. For example, I am doing research at a local hospital and have talked with a number of the residents preparing to go out and establish practices. They were fascinated with the idea of computerizing their records but they don't want to write the

programs or spend time tinkering with their computer's innards. Where oh where is the machine for them?! I would love to have my Poly handle the statistical end of my research but where do I get the software? How do I get that very expensive piece of equipment to work for me like my car does or my TV set or any of the other pieces of complex equipment which I use every day? I would like to hear from some of the manufacturers since it is they who have to make the changes. What do you people think? Do you have any plans to standardize BASIC or to produce computers which are simple enough to go to work immediately and which have the software to make them useful tools?

Nuff said for now.

Annette Ran
17250 Cornell
Southfield MI 48075

FROM ANDREW CLEMENT

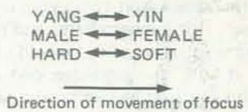
Dear *People's Computers,*

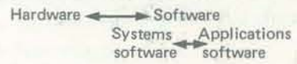
The article 'Women and Computers' raises an important issue - it has too long been neglected and needs to be discussed much more (the same can be said about participation by other groups in computing and in fact the sociological and anthropological aspects of computing are ignored too often). The topic was well-covered, and I found myself agreeing with much that was said. The thing I take most exceptions to is, Don Inman's downplaying the importance being female rather than simply a beginner. Being a beginner is obviously a big part of it and a great deal must be done so that novices, of any flavor, can over-come the barriers to computing more easily. However, apart from this, women face special and significant obstacles to more active participation that need to be addressed particularly. It is important to realize that a female beginner is, *on average,* much more of a beginner than a male, even though they may both have comparable intellectual abilities and education. This is because even from a very early age our culture directs males much more towards mechanisms and the mastery of the physical world. These are the life experiences that Annette mentioned and the skills thus developed happen at a much more basic level than simple 'electricity and machine building.' This orientation starts early and pervades much of a male's upbringing, although there are recent signs that this may be changing in North America.

Even if a woman has overcome these background deficiencies, she will, on average, still have a harder time than a comparable man. The existing male dominance works strongly against a woman trying to enter the field. Firstly, people's expectations are conditioned by what they see around them and so will not as easily accept a woman. In most social situations a person who is obviously different than the rest is at a severe disadvantage unless they make a special effort to overcome the barriers of suspicion and confusion resulting from the mutual lack of familiarity. Thus the social environment in which computing is learned and performed is very important in the development of female computerists. In this I am very much in agreement with Rita Liff's remarks and the action program she describes looks excellent. The four aspects of the program are addressed directly to overcoming the lack of background and developing the confidence and encouragement needed to deal with the current social reality of the male-dominated computer world. I certainly hope the program is successful and gets the support it deserves.

The other side of the coin is for us men to be more aware of our privileged position and do what we can to make 'computers available to the people regardless of race, creed, sex or technological background'. This means that we make a conscious effort to welcome newcomers, try to understand their problems from their point of view and do what we can to help. This will be hard because we are unaccustomed to doing this. But the rewards of a more diverse, stimulating, and egalitarian computing community should make it well worth the effort.

An aside: One question that has not been raised is whether women are inherently (biologically, evolutionarily) less well suited to computing than men? When I showed the women and computers article to a woman friend (who works with computers) she remarked that one reason that more women were not involved is that computers are predominantly 'yang' in nature whereas women tend to be more 'yin'. This started me thinking. I had for some time observed that the few women in computing were predominantly in the applications software end, many fewer in system software and I know of no woman who works mainly with hardware. The further one gets away from the internal workings of the mechanism the greater the presence of women. Even the terms hardware and software correspond to 'yang' and 'yin' (hard - yang/ soft - yin). This speculative analysis levels itself nicely to a series of spectrum diagrams,





The movement of focus arrow shows the general direction in which attention travels. This can be seen in the 'micro revolution'. First came the chips, then the CPU boards, basic system software, and now people are paying more attention to real applications. With each technological innovation attention snaps back to the left and then drifts right as we follow the successive ramifications. What are the consequences for women? Well, yin elements are becoming increasingly involved with computers and are in fact vital to breathe warm life into our technological gadgetry. Computing, taken in a broad view, is not purely yang but spans the spectrum and offers a place for anyone. It is only by the melting of diverse and varied interests that the full potential of computing to serve as a tool for all people who want it can be realized. Otherwise it will remain sterile gimmickry — a peculiar form of mental-mechanical masturbation.

This analysis shouldn't be interpreted as suggesting that hardware should remain male dominated or that there is something unnatural or unhealthy about women wanting to participate in it. Individual variations in background experience and opportunity are so great that each person has to be regarded on their own merits, and making prior judgments about an individual is unfair and discriminatory.

Andrew Clement
789 West 18th Avenue
Vancouver, BC
Canada V5Z 1W1

Computer Aided Instruction in BASIC

BY FRANZ J. FREDERICK

Dr. Franz J. Frederick is an associate professor of Media Sciences in the Department of Education at Purdue University. He teaches courses in Computer Assisted Instruction and Information Science. These courses are centered around the use of micro-computers in the schools. His home computer system is based on a SWTPC 6800 board set with 16 K of RAM, a KC standard 300/600 BAUD tape interface, 128 characters per line 16 line video terminal, and a servo controlled plotting robot named 'Waldo' whose photo appeared in the last issue of People's Computers. *This article is a combination of two articles by Dr. Frederick from the proceedings of the 'First West Coast Computer Faire', it is reprinted with permission.*

INTRODUCTION

Computer assisted instruction lessons can be written in almost any computing language. Why then are there specialized computer-assisted instruction languages? The answer is threefold. One, CAI languages typically have a limited set of commands. Two, CAI languages usually have some special answer processing functions. Three, CAI languages usually provide facilities for recording student performance and for providing a copy of those records for the teacher. Virtually all CAI languages are interactive; those which aren't tend to come under the rubric of computer-managed instruction (CMI).

The major design intent of these languages was centered around the notion of providing certain basic computational capability for use by teachers. It was assumed that these teachers should be able to use the language without being experienced programmers. It was also assumed that the primary use would be tutorial based upon sets of questions and answers.

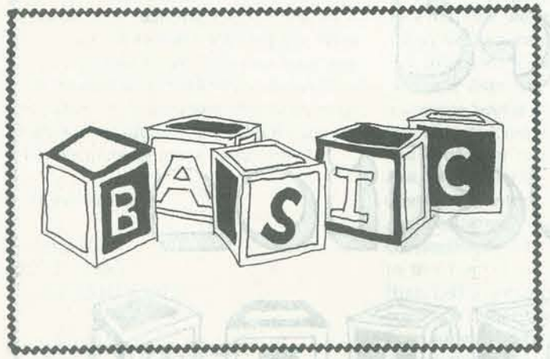
With the advent of low cost micro-processors and in particular the advent of multi-user micro-processor systems, the ideal of low cost CAI in the classroom seems possible. The only generally available higher level language currently on micro systems is of course BASIC in more or less extended versions. While BASIC is a general purpose computing language and does not have the built-in CAI functions, it is possible to generate these capabilities through subroutines.

Recognizing that it may be some time before a micro CAI language is generally available and that extended BASIC with string functions is likely to be a defacto standard, the author designed generalized CAI subroutines in BASIC. Those presented in this paper are keyword and phonetic answer processors and a student records subroutine.

KEYWORD ANSWER PROCESSING

The first efforts at answer processing involved exact answer match. After initial experimentation, teachers began to discover that exact answer matching simply was not effective for many types of tutorial lessons. For example, the student's answer to a question might have the same words in a different order and therefore be counted wrong. The exact answer match implicitly required exact order as well.

This dilemma lead to the development of keyword answer processors. In this case, the author specified a keyword and if that word occurred anywhere in the student's response, the response was considered correct. Embellishments quickly became necessary and included such things as (1) multiple keywords and (2) allowing the keyword to be embedded with a larger word. This latter feature allowed the student to respond with the plural form or past tense of the authors' answer and still be considered correct.



After some further experimentation teachers began to find that exact order of keywords was indeed important and desirable for some lessons. Consequently, the next major embellishment of keyword processors allowed stipulation of exact order or no order. These developments in keyword processors were not only very useful in the standard automated lesson but opened the way for experimentation with simulated conversational interaction. The counselor-patient types of simulations became possible providing the teacher could specify reasonable anticipated questions or question sequence which could reasonably occur in real life situations.

The developments were rather useful in computer-assisted instruction applications but basically were available only in CAI languages. The actual implementation of answer processors usually treats the processors as language functions with parameters.

The generalized keyword subroutine presented here was designed to allow specification by the author of up to five keywords and to allow the author to specify how many must be matched in order to be considered correct.

KEYWORD ALGORITHM

The keyword algorithm is as follows:

1. Compare each author keyword with student's response.
2. If a keyword matches, increment the match counter.
3. Compare the match counter with the authors' specified number of matches. If equal, print a correct response message, record indication of correct response and record the student's actual response. Blank the answer variable. Return to calling routine.
4. If no match in step 3, check to see if all keywords have been compared without success. If so, print an incorrect response message, record indication of incorrect response and record the student's actual response. Blank the answer variable. Return to the calling routine.

CALLING PROGRAM DESIGN

The design of an answer processing routine is dependent to some extent upon the design of the calling program.

Most CAI languages treat lessons as blocks of information to be presented to the student. The block may be composed of (1) information to be displayed, (2) question, (3) specified answers to be compared to a student's response and (4) actions to be performed dependent upon the quality of the students' response.

EXAMPLE OF 'BLOCK'

A suggested block in BASIC would be as follows:

```
500 PRINT "NAME A PROPERTY OF COLOR"
505 REM---STUDENT RESPONSE VARIABLE IS A$ AND IS THE SAME
506 REM----IN ALL BLOCKS
510 INPUT A$
520 REM----R IS NUMBER OF BLOCK IN THE PROGRAM; K IS THE
521 REM---NUMBER OF KEYWORDS TO MATCH
530 R=9 : K=1
540 REM---AUTHOR SPECIFIED KEYWORD
550 A$(1)="HUE"
560 A$(2)="VALUE"
570 A$(3)="INTENSITY"
580 REM---CALL KEYWORD SUBROUTINE
590 GOSUB 7000
```

DESIGN OF KEYWORD SUBROUTINE

The keyword subroutine presented in this article requires the following extended BASIC features:

1. STRINGS (length at least equal to 70 characters)
2. MID$ (A$,I,J)
3. LEN(A$)

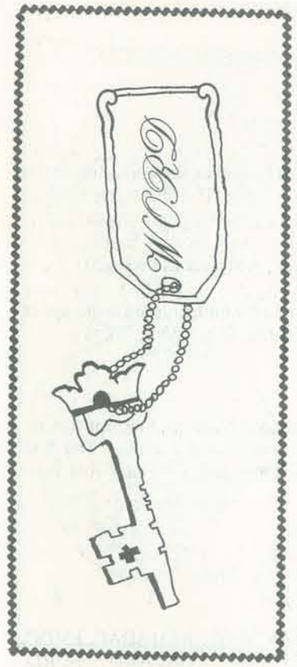
The keyword subroutine requires the following unique variables:

1. K2 (number of matches)
2. K (number of keywords to match—see BLOCK design)
3. K1 (actual number of author keywords)
4. A$ (string variable which holds student answer)
5. R$(I,J) (string matrix used to record student responses and performance record)
6. A(5) (numeric matrix used to hold length of keyword strings)

The keyword routine uses only two temporary variables (I and J). They are used as loop counter variables.

The keyword subroutine allows the following keyword checks:

1. Specification of single keyword match from field of 1 to 5 author specified keywords.
2. The keyword is treated as a root keyword consequently it will allow the occurrence of the keyword with prefix or suffix in the student response.



KEYWORD PROGRAM LISTING

```
 1 REM --- CLEAR VARIABLE SPACE AND ASSIGN 1000 BYTES TO STRINGS
 2 CLEAR 1000
 9 REM --- DECLARE NUMERIC AND STRING MATRICES
 10 DIM A(5),A$(5),R$(10,3)
 95 REM --- FIRST "BLOCK" OF INSTRUCTIONAL MATERIAL
 96 REM     DISPLAY QUESTION AND REQUEST RESPONSE
 97 REM       A$ IS USED AS A GENERAL ANSWER VARIABLE FOR ALL
 98 REM       "BLOCKS"
 100 PRINT:PRINT:PRINT "WHAT IS ONE PROPERTY OF COLOR";:INPUT A$
 105 REM --- THE VARIABLE "R" IS USED TO NUMBER THE BLOCKS AND
 106 REM       IS SUBSEQUENTLY USED AS AN INDEX TO THE STUDENT
 107 REM       RECORD MATRIX "R$(R,C)"
 108 REM     THE VARIABLE "K" IS USED BY THE TEACHER TO SPECIFY
 109 REM     THE NUMBER OF KEYWORDS REQUIRED FOR A CORRECT RESPONSE
 110 R=1:K=1
 115 REM --- THE SUBSCRIPTED STRING VARIABLE A$(N) IS USED BY
 116 REM     THE TEACHER TO HOLD THE KEYWORDS CONSIDERED TO BE
 117 REM     CORRECT ANSWERS
 120 A$(1)="VALUE":A$(2)="HUE":A$(3)="INTENSITY"
 125 REM --- CALL KEYWORD PROCESSOR SUBROUTINE (THE SUBROUTINE
 126 REM       JUDGES CORRECTNESS OF RESPONSE AND AUTOMATICALLY
 127 REM       RECORDS THE INDICATION OF CORRECTNESS IN R$(R,C)
 130 GOSUB 7000
 195 REM --- ANOTHER INSTRUCTIONAL "BLOCK" BEGINS HERE.  THE
 196 REM     STRUCTURE IS THE SAME AS FOR BLOCK 1 (R=1).
 200 PRINT:PRINT:PRINT "NAME TWO PROPERTIES OF COLOR"
 210 INPUT A$
 220 R=2:K=2
 230 A$(1)="INTENSITY":A$(2)="VALUE":A$(3)="HUE"
 240 GOSUB 7000
 295 REM --- "BLOCK 3" BEGINS HERE AND HAS THE SAME STRUCTURE
 296 REM     AS THE PRECEEDING BLOCKS.
 300 PRINT:PRINT:PRINT "NAME ALL THREE PROPERTIES OF COLOR"
 310 INPUT A$
 320 R=3:K=3
 330 A$(1)="HUE":A$(2)="INTENSITY":A$(3)="VALUE"
 340 GOSUB 7000
 495 REM --- THE NEXT STATEMENT CONCLUDES THE BLOCKS SUPPLIED
 496 REM     TO TEST THE KEYWORD PROCESSOR SUBROUTINE.
 500 GOTO 9999
 6994 REM
 6995 REM --- ****** THE KEYWORD PROCESSOR SUBROUTINE ******
 7000 K2=0
 7005 REM --- DETERMINE THE LENGTHS OF THE AUTHORS KEYWORDS
 7010 A(1)=LEN(A$(1))
 7020 A(2)=LEN(A$(2))
 7030 A(3)=LEN(A$(3))
 7040 A(4)=LEN(A$(4))
 7050 A(5)=LEN(A$(5))
 7055 REM --- DETERMINE HOW MANY KEYWORDS WERE SUPPLIED FOR
 7056 REM     TEST BY THE AUTHOR.
 7060 FOR I=1 TO 5
 7070 IF A(I)=0 THEN 7090
 7080 NEXT I
 7085 REM --- THE VARIABLE K1 CONTAINS THE ACTUAL NUMBER OF
 7086 REM     KEYWORDS IN THE AUTHORS LIST
 7090 K1=I
 7095 REM --- COMPARE THE NUMBER OF KEYWORDS SPECIFIED FOR MATCH
 7096 REM     WITH THE NUMBER OF KEYWORDS IN THE LIST.  IF
 7097 REM     K IS GREATER THAN K1 THEN ASSUME ALL IN THE
 7098 REM     LIST MUST MATCH.
 7100 IF K>K1 THEN K=K1
 7110 I=1
 7115 REM --- CHECK TO SEE IF CURRENT KEYWORD BEING TESTED
 7116 REM       IS EMPTY (IE, HAS NO CHARACTERS), IF SO TERMINATE
 7117 REM       THE TEST SEQUENCE.
 7120 IF A(I)=0 THEN 7200
 7125 REM --- COMPARE THE CURRENT KEYWORD WITH ALL SEQUENCES OF
 7126 REM       CHARACTERS IN THE STUDENTS' ANSWER.
 7130 FOR J=1 TO LEN (A$)
 7140 IF MID$(A$,J,A(I))<>A$(I) THEN 7150
 7141 REM --- IF MATCH, INCREMENT MATCH COUNTER
 7142 K2=K2+1
 7144 GOTO 7160
 7150 NEXT J
 7155 REM --- COMPARE MATCH COUNTER WITH THE NUMBER OF KEYWORDS
 7156 REM       SPECIFIED AS NECESSARY FOR CORRECT RESPONSE.  IF
 7157 REM       EQUAL, PRINT POSITIVE MESSAGE FOR STUDENT.  IF NOT,
 7158 REM       CHECK TO SEE IF ALL KEYWORDS IN LIST HAVE BEEN
 7159 REM       HAVE BEEN TESTED.
 7160 IF K2=K THEN PRINT "RIGHT":GOTO 7230
 7165 REM --- IF ALL KEYWORDS IN THE LIST HAVE BEEN TESTED, PRINT
 7166 REM       NEGATIVE MESSAGE TO THE STUDENT.
 7170 IF I=5 THEN 7200
 7180 I=I+1
 7190 GOTO 7120
 7200 PRINT "WRONG!"
 7205 REM --- STORE INDICATION OF WRONG RESPONSE IN THE STUDENT RECORD
 7206 REM     MATRIX.
 7210 R$(R,3)="-"
 7220 GOTO 7240
 7225 REM --- STORE INDICATION OF CORRECT RESPONSE IN THE STUDENT RECORD
 7226 REM     MATRIX.
 7230 R$(R,3)="+"
 7235 REM --- STORE ACTUAL STUDENT RESPONSE IN THE STUDENT RECORD
 7236 REM     MATRIX
 7240 R$(R,2)=A$
 7245 REM --- BLANK RESPONSE VARIABLE.
 7250 A$=" "
 7255 REM --- RETURN TO THE CALLING "BLOCK".
 7260 RETURN
 9999 END
```

PHONETIC ANSWER PROCESSING

Answer processing using keywords represents a very powerful tool in the development of conversational simulations. Programs can be devised which can carry on a useful conversation in a limited context.

While keyword answer processing opened new horizons in the development of tutorial lessons, teachers began to encounter problems with students who could not spell or could not type accurately. These problems lead to a concern for some sort of answer processor which could phonetically encode the responses and thus avoid some basic problems in typographic errors and spelling errors.

A very basic but interesting phonetic answer processor algorithm was developed through a project funded by the National Science Foundation. The project resulted in the development of a CAI language called PLANIT. It was to be "machine-independent". It successfully met that goal with the only restrictions being a 24 bit word size (minimum) and access to a FORTRAN IV compiler. The phonetic algorithm used in PLANIT is quoted verbatim in this article.

BASIC PHONETIC PROCESSOR PROCEDURE

The general algorithm for a phonetic answer processor in BASIC is as follows:

1. Disassemble the answer or response string into single character strings.
2. Sequentially convert each single character string into its' phonetic equivalent using the PLANIT PHONETIC ALGORITHM.
3. Reassemble phonetic characters equivalents into a single string.
4. Return to calling program for comparison of phonetic equivalent response to a phonetic equivalent answer specified by the lesson author.

PHONETIC ENCODING AND FORMULAS PROCESSING*

The phonetic answer processor subroutine requires the following extended BASIC features:

1. STRINGS (length at least equal to 70 characters)
2. MID$(A$,J,1)
3. LEN(A$)

PHONETIC ENCODING

The phonetic encoding process is accomplished in four steps:

Step 1 - Letter Equivalent:

All letters are transformed into their letter equivalents. Any remaining characters including blanks are unchanged. The letter in Row 1 is transformed into the letter immediately below in Row 2. PLANIT ignores all other characters.

Row 1 ABCDEFGHI JKLMNOPQRSTUVWXYZ
(original letter)
Row 2 ABCDABCHACCLMMABCRCDABHCAC
(letter equivalent)

Step 2 - The H Replacement:

Each H in a word is transformed to the preceding letter provided the character is a letter. If not a letter (e.g. , a blank), H is unchanged.

Step 3 - Elimination of Successive Identical Consonants:

All but the first element of an uninterrupted sequence of a single consonant is eliminated, (e.g., CC=C, TT=T).

Step 4 - Elimination of A's:

All vowels, transformed A's, are eliminated except if A is the first character of the word to be encoded. The final word contains only consonants and a leading A if there is one.

Examples:

Original Word	Steps			
	1	2	3	4
PHONETIC	BHAMADAC	BBAMADAC	BAMADAC	EMDC
HAZARD	HACARD	HACARD	HACARD	HCRD
ON–LINE	AM–LAMA	AM–LAMA	AM–LAMA	AM–LM
AWHILE	AHHALA	AAAALA	AAAALA	AL

The phonetic subroutine requires the following unique variables:

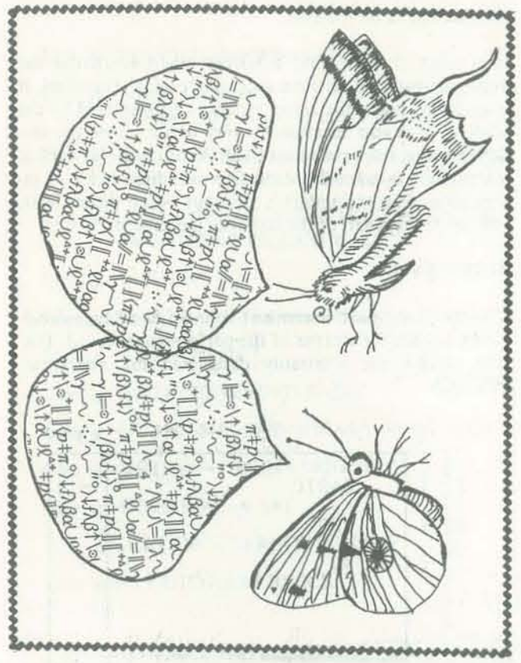
1. B$ (phonetic encoded answer)
2. A$ (string variable to hold student response)
3. R$(I,J) (string matrix to hold student performance records)
4. B$(73) (string matrix used to hold individual characters from student response or author answer)

The phonetic subroutine uses only two temporary variables (J and K). They are used as loop counter variables and index pointers for matrices.

The phonetic answer processor subroutine is designed to be called to process both the authors' answer and the students' response (on separate GOSUB calls). The phonetic response processor can handle character strings up to 70 characters length with embedded blanks and punctuation. The only restriction is that the use of a comma in the students' answer may produce a syntax error depending on which BASIC interpreter you use.

The listing of a sample mini-lesson (one "block") and the phonetic subroutine follow.

*Bennick, F.D. and C.H. Frye, ***PLANIT LANGUAGE REFERENCE MANUAL,*** System Development Corporation TM-(L) -4422/002/01, Oct. 1970. (APPENDIX E)



```
8020 B$(K)=MID$(A$,J,1)
8022 REM ~~~ TRANSFORM ANSWER LETTERS INTO LETTER EQUIVALENTS.  BLANKS
8024 REM     REMAIN UNCHANGED.  LETTERS TRANSFORMED AS FOLLOWS:
8026 REM
8028 REM     ORIG.  A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
8029 REM     EQUIV. A B C D A B C H A C C L M M A B C R C D A B H C A C
8030 IF B$(K)="E" OR B$(K)="I" OR B$(K)="O" THEN B$(K)="A"
8040 IF B$ (K)="U" OR B$(K)="Y" THEN B$(K)="A"
8050 IF B$(K)="F" OR B$(K)="P" OR B$(K)="V" THEN B$(K)="B"
8060 IF B$(K)="G" OR B$(K)="J" OR B$(K)="K" THEN B$(K)="C"
8070 IF B$(K)="Q" OR B$(K)="S" THEN B$(K)="C"
8080 IF B$(K)="X" OR B$(K)="Z" THEN B$(K)="C"
8090 IF B$(K)="W" THEN B$(K)="H"
8100 IF B$(K)="N" THEN B$(K)="M"
8110 IF B$(K)="D" THEN B$(K)="D"
8115 REM ~~~ TRANSFORM H INTO PREVIOUS LETTER UNLESS IT BEGINS WORD
8120 IF B$(K)="H" AND B$(K-1)<>" " THEN B$(K)=B$(K-1)
8130 IF B$(K)="A" THEN 8150
8135 REM ~~~ REPEATED CONSONANT DELETED
8140 IF B$(K)=B$(K-1) THEN B$(K)="";K=K-1
8145 REM ~~~ REPEATED A DELETED
8150 IF B$(K)="A" AND B$(K-1)="A" THEN B$(K)="";K=K-1
8155 REM ~~~ DELETE A AFTER CONSONANTS
8160 IF B$(K)="A" AND B$(K-1)<>" " THEN B$(K)="";K=K-1
8165 REM ~~~ A AFTER A BLANK ALLOWED (EG AT START OF A WORD)
8170 J=J+1
8180 K=K+1
8190 GOTO 8010
8200 FOR M=1 TO LEN(A$)
8220 B$=B$+B$(M)
8230 NEXT M
8240 RETURN
9999 PRINT:PRINT "******** END OF PHONETIC SUBROUTINE TEST **********":END
```

PHONETIC RESPONSE PROCESSOR LISTING

```
 1 REM ~~~ CLEAR VARIABLE SPACE AND ASSIGN 500 BYTES TO STRINGS
 2 CLEAR 500
 9 REM ~~~ DECLARE STRING MATRICES FOR PHONETIC SUBROUTINE
10 DIM B$(73)
15 REM ~~~ DECLARE STRING MATRIX FOR STUDENT PERFORMANCE RECORDS
16 REM     THE R DIMENSION REFERS TO THE NUMBER OF "BLOCKS"
17 REM     ENCOUNTERED BY THE STUDENT.  THE C DIMENSION REFERS
18 REM     TO (1) BLOCK NUMBER, (2) ACTUAL RESPONSE, AND (3)
19 REM     THE RESULT OF COMPARING RESPONSE WITH ANSWER
20 DIM R$(30,3)
70 B$(0)=" "
80 I=0
95 REM ~~~ FIRST "BLOCK" OF INSTRUCTIONAL MATERIAL.
96 REM     PRESENT QUESTION - -
100 PRINT "WHO WAS THE FIRST PRESIDENT OF THE U.S."
105 REM ~~~ THE VARIABLE R REFERS TO THE BLOCK NUMBER.
106 REM     THE VARIABLE I IS USED TO INDICATED THE NUMBER OF
107 REM     ATTEMPTS IN THE LESSON.  THE MATRIX R$(R,C)
108 REM     IS USED TO HOLD A RECORD OF THE STUDENTS'
109 REM     PERFORMANCE BLOCK BY BLOCK.
110 R=1:I=I+1:R$(I,1)=STR$(R)
115 REM ~~~ A$ IS USED TO HOLD THE AUTHOR SPECIFIED ANSWER
120 A$="WASHINGTON"
125 REM ~~~ GOTO PHONETIC SUBROUTINE AND CONVERT ANSWER TO
126 REM     PHONETIC REPRESENTATION.  RETURN IT IN B$.
130 GOSUB 8000
135 REM ~~~ MOVE PHONETIC REPRESENTATION FROM TEMPORARY VARIABLE
136 REM     B$ TO P$.
140 P$=B$
145 REM ~~~ REQUEST STUDENT RESPONSE TO THE QUESTION.
146 REM     A$ IS AGAIN USED TO HOLD DATA TO BE PASSED TO
147 REM     PHONETIC SUBROUTINE.
150 INPUT A$
155 REM ~~~ STORE STUDENTS' ACTUAL RESPONSE IN RECORD MATRIX
160 R$(I,2)=A$
165 REM ~~~ GOTO PHONETIC SUBROUTINE AND CONVERT RESPONSE TO
166 REM     REPRESENTATION.  RETURN IT IN B$.
170 GOSUB 8000
175 REM ~~~ COMPARE B$ WITH P$.
180 IF P$<>B$ THEN 210
185 REM ~~~ RECORD CORRECT RESPONSE INDICATOR
190 R$(I,3)="+"
195 REM ~~~ PRINT FEEDBACK MESSAGE TO STUDENT
200 PRINT "RIGHT!";GOTO 230
205 REM ~~~ RECORD INCORRECT RESPONSE INDICATOR
210 R$(I,3)="-"
215 REM ~~~ PRINT FEEDBACK MESSAGE TO STUDENT
220 PRINT "WRONG!"
225 REM ~~~ PROCEED TO NEXT FRAME
229 REM *** END OF TEST OF PHONETIC SUBROUTINE
230 GOTO 9999
7996 REM ******** PHONETIC SUBROUTINE **********
7998 REM ~~~ SET UP VARIABLES FOR COUNTERS
8000 K=1:J=1
8002 B$=" "
8005 REM ~~~ PLACE EACH LETTER/CHAR IN ANSWER INTO SEPARATE CELL
8010 IF J>LEN(A$) THEN 8200
```

STUDENT RECORDS

There are three major purposes for maintaining records of student performance. One, the learning process is enhanced when the learner knows immediately of the quality of his performance. It is also of particular value to the learner to know of his performance relative to the total task or lesson. Two, the teacher can more effectively guide the student in learning if the teacher has records which indicate the student's performance with respect to specific items. Three, the teacher can use performance records to assess the effectiveness of the learning materials – i.e., the teacher can identify those areas of the task needing re-design.

The first purpose described above can be accomplished by the use of simple correct and incorrect answer counters of the form:

```
line # R=R+1          (Right Answer)

or

line # W=W+1          (Wrong Answer)
```

The appropriate counter is incremented after judging the student's response. At the end of the program, the lesson author would probably use something similar to the following code:

```
Line # PRINT "YOU GOT ";R;" RIGHT WITH ";W;
       "WRONG!"

Line # PRINT "YOUR OVERALL PERFORMANCE
       WAS";INT((R/R+W)*100);% CORRECT."
```

The procedure just described is entirely student oriented, ie, no records are kept for teacher use. The second and third purpose for maintaining performance records is to make information available for teachers.

STUDENT RECORDS DESIGN

The design of a more pervasive student records procedure may be summarized as follows:

1. Identify the information to be retained.
2. Specify matrix layout required to maintain the information.
3. Identify items to be manipulated or defined for each question/answer block presented.
4. Specify information to be presented in a summary report form.
5. Specify completed record report form.

The design presented here is predicated on a machine with no disk or a machine with a BASIC which permits storage of array/matrix data.

1. Information to be retained.

It is often possible that a learner might encounter the same question/answer item more than once. Therefore, it is useful to know the actual sequence encountered by the student. It is also occasionally very useful to see the student's actual response to an item. Another useful item of information is some identification of which block of information was encountered. Last but by no means least, one needs the result of the response judgement.

2. Matrix layout.

It is probably most convenient to use a two dimensional string matrix for storage of the performance record. The The author has arbitrarily designated the matrix as R$(R,C).

		1	2	3
	1	1	BASIC	+
	2	2	YES	-
	3	2	NO	+
ROWS	4			
	5			
	6			
	7			
	n		$QUIT	

EXAMPLE OF RECORDS MATRIX

3. Items to be defined or manipulated in a block.

Most CAI languages treat lessons as discrete "chunks" of information. The "chunks" may be either information to be printed or questions and answers or a combination. We will refer to such an arrangement as a "block".

The structure of a block would appear as follows:

```
Line # R=n (Defines number of block)
Line # I=I+1 (The Ith time the student has encountered
       an item; used as a row index to records matrix)
Line # R$(I,1)–STR$(R) (Stores a string representation
       of the block number in the record matrix)
Line # PRINT "WHAT IS THE MOST AVAILABLE MI-
       CRO COMPUTER LANGUAGE"
Line # INPUT A$
Line # R$(I,2)=A$ (Store student answer in matrix)
Line # A$(1)="BASIC"
Line # A$(2)="EXTENDED BASIC"
Line # IF A$◊A$(1) AND A$◊A$(2)THEN LINE  #n
Line # R$(I,3)="+":PRINT "RIGHT!"
Line # GOTO LINE  # n+1
Line #n R$(I,3)="–":PRINT "WRONG!"
Line #n+1 GOTO NEXT BLOCK
```

Note: After last block, Set I=I+1 THEN R$(I,2)= "$QUIT"
(This signals the end of the entries in the matrix)

4. Summary report.

The summary report should print a report heading and the columnar headings - "block number" and "judgement". It should also print the total number correct, total number incorrect, and a performance percentage.

5. Complete report.

The complete report should include a column for the actual response as well as all the data for the summary report.

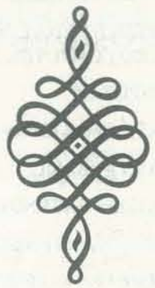
The subroutines are actually only the routines used to print the contents of the records. The records are actually generated through the blocks. The subroutines are executed after the student completes his lesson by typing GOTO n where n is the starting line number for the desired report.

The following listing shows the subroutines and two example blocks.

STUDENT RECORDS PROGRAM LISTING

```
10 CLEAR 500
50 DIM A$(5)
60 DIM R$(100,3)
80 PRINT:PRINT:PRINT "     CAI LESSON AND DEMONSTRATION OF STUDENT RECO
RDS
   "
90 PRINT:PRINT
100 INPUT "WHAT IS YOUR NAME";N$
110 I=0
115 REM ******** FIRST BLOCK IN LESSON *********
120 R=1:I=I+1:R$(I,1)=STR$(R)
130 PRINT "NAME THE MOST POPULAR LANGUAGE ON MICRO-COMPUTERS CURRENTLY"
140 INPUT A$
150 A$(1)="BASIC"
160 A$(2)="EXTENDED BASIC"
170 A$(3)="MITS BASIC"
180 A$(4)="BK BASIC"
190 R$(I,2)=A$
200 IF A$<>A$(1) AND A$<>A$(2) and A$<>A$(3) AND A$<>A$(4) THEN 250
210 R$(I,3)="+"
220 GOTO 270
250 R$(I,3)="-"
260 REM ****** SECOND BLOCK IN LESSON **********
265 REM --- NOTE THAT THIS BLOCK REQUIRES THE STUDENT TO GET THE ANSWER
266 REM     CORRECT BEFORE HE CAN MOVE AHEAD IN THE LESSON.
270 R=2:I=I+1:R$(I,1)=STR$(R)
280 PRINT "IS IT AVAILABLE IN ALL MICRO COMPUTERS"
290 INPUT A$
300 A$(1)="NO"
310 R$(I,2)=A$
320 IF A$=A$(1) THEN R$(I,3)="+":GOTO 350
330 PRINT "TRY AGAIN:"
340 R$(I,3)="-":GOTO 270
350 GOTO 9995
9000 REM --- STUDENT RECORD PRINT SUBROUTINES
9002 REM     (1) SUMMARY FORM (INCLUDES BLOCK NUMBER AND RESPONSE
9003 REM         INDICATOR)
9004 REM     (2) COMPLETE STUDENT RECORD (BLOCK NUMBER, ACTUAL
9005 REM         STUDENT RESPONSE, AND RESPONSE INDICATOR)
9010 PRINT:PRINT:PRINT "****** STUDENT PERFORMANCE RECORD FOR ";N$;" ***
***"
9020 PRINT TAB(18);"(SUMMARY FORM)"
9030 PRINT
9040 PRINT "BLOCK NUMBER";TAB(15);"ANSWER JUDGEMENT"
9045 REM --- R IS THE VARIABLE USED TO TALLY THE CORRECT RESPONSES.
9046 REM     W IS THE VARIABLE USED TO TALLY THE INCORRECT RESPONSES.
9047 REM     N IS THE VARIABLE USED TO COUNT THE TOTAL RESPONSES.
9050 PRINT:R=0:W=0:N=0
9055 REM --- MAIN LOOP USED TO EXAMINE STUDENT PERFORMANCE ITEM BY ITEM
9056 REM     INFORMATION EACH ITEM IS PRINTED ON A LINE.
9057 REM     WHEN "$QUIT" IS FOUND IN THE ACTUAL ANSWER AREA FOR THE
9058 REM     LAST ENTRY IN THE RECORDS MATRIX, THE PROGRAM THEN
9059 REM     PRINTS TOTALS AND PERCENTAGE OF PERFORMANCE.
9060 FOR I=1 TO 100
9070 IF R$(I,2)="$QUIT" THEN 9120
9075 REM --- PRINT ITEM (BLOCK) NUMBER AND RESPONSE INDICATOR.
9080 PRINT TAB (5);R$(I,1);TAB(20);R$(I,3)
9085 REM --- COUNT CORRECT RESPONSE
```

```
9090 IF R$(I,3)="+" THEN R=R+1
9095 REM --- COUNT INCORRECT RESPONSE
9100 IF R$(I,3)="_" THEN W=W+1
9110 NEXT I
9115 REM --- SET N COUNTER TO ONE LESS THAN I BECAUSE LAST ITEM IN
9116 REM     STUDENT RECORD IS USED FOR FLAG FOR END OF RECORD.
9120 N=I-1
9130 PRINT
9140 PRINT "TOTAL ANSWER BLOCKS SEEN:     ";N
9150 PRINT "TOTAL INCORRECT ANSWERS:      ";W
9160 PRINT "TOTAL CORRECT ANSWERS:        ";R
9170 PRINT "       PERFORMANCE PERCENTAGE:    ";INT(R/N*100);"%"
9180 PRINT
9190 PRINT "_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
9200 PRINT:PRINT:STOP
9300 PRINT:PRINT:PRINT "****** STUDENT PERFORMANCE RECORD FOR ";N$;" ***
     ****"
9310 PRINT TAB (18);"(COMPLETE RECORD)"
9320 PRINT
9325 REM --- PRINT BLOCK NUMBER, ACTUAL RESPONSE AND ANSWER JUDGEMENT F
9326 REM     EACH ITEM
9330 PRINT "BLOCK NUMBER";TAB(15);"ACTUAL RESPONSE";TAB(50);"JUDGEMENT"
9340 PRINT:R=0:W=0:N=0
9350 FOR I=1 TO 100
9360 IF R$(I,2)="$QUIT" THEN 9410
9370 PRINT TAB (5);R$(I,1);TAB(15);R$(I,2);TAB(55);R$(I,3)
9380 IF R$(I,3)="+" THEN R=R+1
9390 IF R$(I,3)="_" THEN W=W+1
9400 NEXT I
9410 N=I-1
9420 PRINT
9430 PRINT "TOTAL ANSWER BLOCKS SEEN:     ";N
9440 PRINT "TOTAL INCORRECT ANSWERS:      ";W
9450 PRINT "TOTAL CORRECT ANSWERS:        ";R
9460 PRINT "       PERFORMANCE PERCENTAGE:    ";INT(R/N*100);"%"
9470 PRINT
9480 PRINT "_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _"
9490 PRINT:PRINT:STOP
9995 I=I+1
9996 R$(I,2)="$QUIT"
9999 END
```



****

Continued from page 23. . .

Spencer, Cornelia. *Keeping Ahead of Machines: The Human Side of the Automation Revolution.* John Day, 1965. (gr. 6-9)

Srivastava, Jane Jonas. *Computers.* Crowell, 1972. 32p. (gr. 1 - 5)

Steinberg, Fred. *Computers.* Frankin Watts, Inc. 1969. 89p. (gr. 4-6)

Strong, E.L. "How Streams of Water Can Be Used to Create Analogues of Electronic Tubes and Circuits" *Scientific American,* 207:128 (August 1962)

Thomas, Shirley. *Computers.* Holt, Rinehart and Winston, 1975.

Vorwald, Alan and Frank Clark. *Computers: From Sandtable to Electric Brain,* 3rd ed. McGraw, 1970. (gr. 5 and up)

MISCELLANEOUS FICTION ABOUT COMPUTERS

Christopher, John. *The White Mountains.* MacMillan, 1967. (gr. 4-9)

Del Ray, Lester. *The Runaway Robot.* Westminster, 1965. (gr. 4-9)

Fairman, Paul W. *The Forgetful Robot.* Holt, Rinehart and Winston, 1968. (gr. 5-8)

Hayes, William. *Hold That Computer!* Atheneum, 1968.

Key, Alexander. *The Incredible Tide.* Westminster.

Norton, Andre. *The Beast Master.* Harcourt Brace, 1963. (gr. 6-12)

———. *Catseye.* Harcourt Brace and World, 1963. (gr. 6-12)

———. *Ice Crown.* Viking, 1970. (gr. 6-12)

———. *Outside.* Walker, 1974. (gr. 5-12)

———. *The Stars Are Ours!* World, 1954. (gr. 6-12)

Philbrook, Clem. *"Ollie's Team and the Football Computer"* Hastings, 1968. (gr. 4-6)

Steadman, Ralph. *Little Red Computer, Steadman.* McGraw, 1969.

Sundh, Kerstin. *Augusta, Can Do Anything.* Putnam, 1974.

ANNOUNCEMENTS

HARDWARE

THE PET!

The PET Computer by Commodore will be rolling off the assembly in early September at a bargain price of $595. We hear the $100 price raise over the $495 estimate given at Computer Faire will go towards advertising - Commodore is making a serious bid to corner the mass market. Although sales clerks don't yet know about it, it appears Macy's, Sears, & J.C. Penney will carry the PET.

The 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 system, described as 'quite portable', weighs 44 pounds. It 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 keyboard, a major source of complaint by people who've seen the system will be upgraded from the calculator type keyboard shown on prototypes. Just what directions that upgrade will take are unknown; as of now the keyboard is upper case only; a calculator style numeric keypad is included. 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. Sample software includes a card game (cards displayed) a rocket ship landing, and a backgammon board.

The PET's expanded 8K BASIC contains strings, integers and multiple dimension arrays. It has high precision (10 significant digits) and direct memory access through PEEK and POKE. Commodore claims its BASIC is an upward expansion from current popular versions of BASIC, and that it's 20% faster than most other 8K BASICs.

We'll be publishing an interview with Charles Peddle, father of the PET, in our next issue. For those wishing to order a system, orders will be filled in 90 days and are FOB Palo Alto, or money will be refunded.

Commodore International Ltd.
901 California Ave.
Palo Alto, CA 94304
(415) 326-4000

SWTPC CT-64

The Southwest Technical Products Corporation CT-64 Terminal System kit along with the optional CT-VM video monitor is a complete package providing everything needed for a complete stand alone terminal system compatible with modems and ASCII computer systems of every kind.

The kit features 16 lines of 32 or 64 characters per line, scrolling or page mode operation, upper and lower case characters, reversed character printing, control character printing, cursor control and complete control character decoding.

The kit includes the power supply, keyboard, serial interface, beeper, assembly instructions, chassis and cover and is sold in kit form only for $325.00 ppd. in US. The optional CT-VM video monitor is sold assembled, requires the CT-64's power supply and sells for $175.00 ppd.

Southwest Technical Products Corp.
219 W. Rhapsody
San Antonio, Texas 78216
(512) 344-0241

X-Y PLOTTERS

SYLVANHILLS LAB announces the availability of 8080 based software to control their series of plotters. This enables the micro-computer to act as the controller for the plotter and requires about 2K of memory. The software format is such that it may be used in conjunction with application routines available from MICRO-ViSIONS INC. 4926 Travis, Houston, Texas 77002.

Plotters are shipped completely assembled and tested, but require the purchaser to mount them on his drawing surface and do the interconnection between the control PC boards and his computer. This requires an 8 bit parallel I/O port and 5 and 24 volt power sources.

Applications include architectural, mechanical and schematic drawing; PC board artwork; positioning of small objects; computer generated art; games; and many others. Sizes available are 11 x 17 ($750), 17 x 22 ($895) and 22 x 34 ($1200).

SYLVANHILLS LAB. INC.
1 Sylvanway
Box 239
Strafford MO 65757
(417) 736-2664

DOUBLE DENSITY MICROFLOPPIES

The first flexible disk drive for 5¼ inch diskettes to offer double density recording of 250,000 bytes on each side of a diskette, was introduced at the National Computer Conference by Pertec Computer Corporation.

The new drive, Pertec Model FD200 Microfloppy™, is mechanically and functionally interchangeable with the diskette drive introduced earlier by

Shugart Associates but permits writing on 40 tracks versus a 35-track limit for the Shugart SA400 unit, in addition to the double density feature. The Pertec FD200 also allows recording on both sides of a diskette, whereas other units are limited to one side, according to Pertec.

Both the signal interface connector and the dc power connector are compatible with the Shugart equipment, and mounting holes and outline dimensions are the same.

The new FD200 Microfloppy™ disk drive is marketed to OEM users by Pertec and will be incorporated in both OEM and personal computing products offered by Pertec's Microsystems Division. Delivery of sample quantities is slated for August 1977 with production quantities expected in September. Unit price of the FD200 is $405.

Pertec Computer Corporation
21111 Erwin Street
Woodland Hills CA 91367
(213) 999-2020

CENTRONICS MICROPRINTER

Centronics Data Computer Corp. introduced a compact, high speed, low cost microprinter, the Micro-1, at the 1977 National Computer Conference. The microprinter produces copy on aluminum coated paper by discharging an electric arc to penetrate the coating, which is less than one micron thick. Toners and ribbons are not required. The printed characters, unlike those resulting from thermal printing, are impervious to light, temperature, and humidity. In addition, the finished printed page may be reproduced on most office copy machines.

Aimed at the home, hobby, and microprocessor markets, the Centronics microprinter has a higher print speed (240 characters per second) and lower selling price ($595) than competitive models. Initial deliveries are slated to occur during the last calendar quarter of 1977.

Centronics Data Computer Corporation
Hudson, NH 03051
(603) 883-0111

NATIONAL'S GAMES

The Quiz Kid Racer is National's calculator programmed to challenge youngsters in math games against themselves -- or a competitor when the double set is purchased. The degree of difficulty of math questions can be increased by entering the request on the key. The Quiz Kid Racer will carry a suggested retail price of $21.95 as a single unit or $39.95 in the double, competitive model.

National's newest living color video game, Adversary Model 600 offers 23 games to play (action type Pinball, Wipe-out, etc.). Adversary 600 will carry a suggested retail price of $79.95.

The 600 lets players hear the realistic sound as playing surfaces are struck with ball or puck adding to the excitement. Depending upon their skill, players can select three paddle sizes to test their skills. This provides skilled players the opportunity to inflict "English" on their opponents.

The 600 has remote game selection, individual controllers, easy-to-read scoring on the TV screen and is designed for permanent installation.

Scott Brown
National Semiconductor Corporation
Consumer Products Division
1177 Kern Avenue
Sunnyvale, California 94086
(408)733-2600

SOFTWARE

6502 ASSEMBLER, TINY BASIC ON ROMS

Microcomputer Associates Inc. is selling their 6502 Resident Assembler Program (RAP) and Tiny BASIC interpretive program on ROM chips. Two 2K x 8 ROMs comprise the software ROM package housing the 1.75K Resident Assembler and the 2.2K Tiny BASIC program.

RAP generates a listing and places object code into RAM for immediate execution; minimum of 4K x 8 RAM memory is needed with the users' 6502 microcomputer. Tiny BASIC, a subset of Darmouth BASIC, permits immediate entry and execution of Tiny BASIC language programs. Statements include a user subroutine that allows branching, with arguments to assembly language subroutines. ROM software has been designed so that most any I/O devices can be used. The ROMs are totally pin-compatible with 2708-type PROMs.

The RAP/Tiny BASIC ROM package (SW101) is priced at $200 and includes full documentation with deliveries from stock to 30 days ARO. RAP is also available on a set of seven 1702A PROMs (SW200) for $295. Tiny BASIC is available either in paper tape format (SW300) for $25 or on a set of nine 1702A PROMs (SW201) for $275. All software is fully documented with deliveries from stock to 30 days ARO.

Darrell Crow
Microcomputer Associates
2589 Scott Blvd.
Santa Clara CA 95050
(408) 247-8940

MINI WORD PROCESSING

The Software Store has released a Mini Word Processing System running on MITS Altair equipment under Disk Extended Basic for $150.00. Mini Word Processing is designed to help an operator generate letters, text and mailing labels or envelopes. The System consist of seven programs which are driven by a menu select routine from which any of the seven processing programs can be utilized. Each program interacts with the operator to establish file names and drive numbers. The options are selected by the operator using simple Y or N (Yes or No) responses to the detailed program prompts. After each function is completed the System reloads the menu routine. The 51-page User's Manual provided with the System includes detailed instructions concerning all operator prompts, system error messages, a number of examples, and programming considerations for custom applications.

The Software Store
706 Chippewa Square
Marquette MI 49855
(906) 228-7622

CYPHERS

General George Brown, chairman of the Joint Chiefs of Staff, has said with respect to dissent over government surveillance,

> "If any citizen of this country is so concerned about his mail being read or is concerned about his presence in a meeting being noted, I'd say we ought to read his mail and we ought to know what the hell he has done."

Read this Mister Brown,

> dzeyx wtpcw aouyq oudpq etrnp
> wvuwx pnejx pcfqk czlqu lacac

And if you would like to produce such messages that will baffle the codebreakers of the CIA, just send for the CRYPTOGRAPHIC program that will let you easily ENCODE and/or DECODE your private correspondence for maximum security using your versitile home computer system.

Documentation of cypher technique, program listing in BASIC for only $6. Additional $2 for punched paper tape.

Jon Stedman
1528 Summit Rd.
Berkeley, CA 94708

GATHERINGS

BOSTON, AUG 25-27

'Computermania', a brainchild of *Kilobaud's* Wayne Green will be at Boston's Commonwealth Pier Aug 25-27. Yes, that's right, at the same time as the PC '77 show in Atlantic City (see next announcement).

See dozens of microcomputer systems on display and running . . . sit down and give them a try . . . find out why people get hooked on Star Trek . . . find out why 100,000 computermaniacs have gone nuts over microcomputers. See a couple of hundred exhibits of computers, memory boards, printers, floppy disks!

See manufacturers show and tell about their systems . . . and answer your questions – in detail. You don't have to be a computer expert to find out how exciting microcomputers are . . . and why they are going to be a multi-billion dollar business before long. One look will convince you.

See Morse code translated into print . . . even into voice . . . all by microcomputer . . . and at a price within reasonable hobby limits. See Oscar data computers . . . repeater control computers . . . all sorts of fantastic ham applications of microcomputers.

Don't miss the fun at the Pier in Boston . . . where calculators . . . TV Games . . . Microcomputer . . . Hobby computer systems . . . ham computer systmes . . . and even small business computers you may be able to use in your business will be on display and running for you to try out.

Advance tickers are $10 until August 1; after that the price is $12.

Computermania Tickets
Peterborough NH 03458
or call toll free (800) 258-5473

ATLANTIC CITY, AUG 27-28

PC '77 offers you the most complete show of its kind ever held. Proven in '76 and acclaimed in '77 by all the major professional publications as the coming event of the year, this show is a 'must'. Make plans now to attend. Here are some of the scheduled events:

Pre-Convention Professional Seminars
August 22-26 Technical Design Labs and Trenton State College Z80 Seminars at near-by Trenton State College. Five software and four hardware seminars.
August 25, 26, 28 Sybex Seminars at the Shelburne Hotel. Three intensive seminars: Introduction to Microprocessors, Programming Microprocessors, Microprocessors Applications.
August 24, 25, 26 Tychon Inc. Microcomputer Interfacing Workshop at the Shelburne Hotel:
August 26, 27 Osborne & Associates Microprocessors – Where they came from and where they are going, an analysis of all products on the market today. At the Shelburne Hotel.

More New Products Than Ever!
All the products you've been reading about in the ads will be on display at PC '77. Many companies will be showing exciting new products. Heath Company will display exclusively, for the first time, their complete computer line. Solid State Music, Polymorphic Systems, The Digital Group, Thomas Instrumentation, Mos Technology, Technical Design Labs, Southwest Technical Products, Cromemco, E & L Instruments, The Interpring Group, Kent-Moore Instruments, Persci Inc, George Risk Industries, Mid West Scientific, Osborne and Associates, Expandor, Quay Corp, Matrix Publishers, Camelot Publishing Co, Hayden Book Co, Gaw Electronics, Enclosure Dynamics and Soroc Terminals will all be showing new products. Plan to attend!

Free Seminars, Forums, Technical Talks On Heathkit products, micros in medicine, robots, music, ham radio applications, applications of micros for the handicapped, and more!

Before August 10, registration is $8; at the door, it's $10 for the weekend. Make checks payable to 'Personal Computing '77' and send to:

PC '77
Route 1
Box 242
Mays Landing NJ 08330

HOUSTON, SEPT 17-18

FINALLY! AN EXHIBITION IN TEXAS DEDICATED TO HOBBY COMPUTING!

WHAT: Houston Personal Computing Faire

WHERE: Hall of Exhibits, Shamrock Hilton Hotel; So Main at Holcombe, Houston Tx

WHEN: Saturday, September 17, 1977 9am - 6pm
Sunday, September 18, 1977 9am - 4pm

HOW MUCH: $2 per person for all events on both days

WHAT TO SEE:

- Exhibits by computer hobbyists of home systems
- Exhibits by manufacturers of the latest in microcomputing equipment
- Computer games arcade
- Computer chess tournament
- Door prizes
- Lessons for laymen
- Computer generated artwork
- Classes for small businessmen
- Workshops for hobbyists

Houston Personal Computing Faire
P.O. Box 36584
Houston TX 77036

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

BOSTON, OCT 11-14

The 16th Annual Conference of the North American Simulation and Gaming Association will be held at the Park Plaza Hotel in Boston, Massachusetts on October 11 through 14, 1977. The theme of the conference is "Adult and Continuing Education in Simulation and Gaming".

The basic goal of the North American Simulation and Gaming Association (NASAGA) is to advance an optimal, RESPONSIBLE application of the technique of simulation and gaming. The objectives of the Association are:

- to facilitate communication among persons interested in the field of simulation and gaming;
- to promote the training of specialists in the field of simulation and gaming;
- to facilitate communication between these specialists and policy-makers, students, and other concerned persons; and
- to promote the development of better techniques in the field of simulation and gaming.

Special activities being planned in conjunction with the conference include: mobile workshops, optional tours, and a Game Fair at the popular Boston Globe Book Festival, among many others.

NASAGA
c/o Barry R. Lawson
Room 205, Metropolitan College
Boston University
755 Commonwealth Ave.
Boston MA 02215

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

SAN FRANCISCO, OCT 18-20

'The Industrialization of Space' is the focus of the 23rd annual meeting of the American Astronautical Society, to be held at the Airport Hilton Hotel, San Francisco Airport, October 18-20. Among the topics to be discussed will be
Technical (including communications, navigation, and manufacturing – which may require large space structures and space settlements)
Space Law
Community Planning in Space
Psycho-Social Aspects of Living and Working in Space
Economic Realities

The conference will bring together these various aspects in an integrated manner, to give a fuller understanding of the problems facing us as we look to Space for future profit; and to bring a greater appreciation of the benefits awaiting us at this new frontier. The conference will help assure that as each step becomes technologically feasible, it also becomes financially feasible.

The meeting is co-sponsored by a variety of organizations, including Stanford Research Institute, IEEE, and the L-5 Society. Special tours of facilities dealing with Space have been set up by local organizations. A few such tours include Stanford Linear Accelerator, NASA/Ames, and the Exploratorium.

American Astronautical Society
P.O. Box 7205
Menlo Park CA 94025
Paul L. Siegler (415) 494-8339
E.V. Stearns (408) 742-8150

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

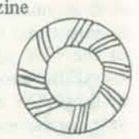
CHICAGO, OCT 27-29

Another 'Personal Computing' show, October 27-29, at the Holiday Inn at Chicago's O'Hare International Airport. The show will feature a variety of personal computer systems, new products, homebrewed systems and applications – all of it directed at the computer neophyte. Manufacturers and distributors will be offering consumer discounts for cash purchases at the show – some up to 50%! Door prizes, grand prizes, gifts and surprises of special interest to computer hobbyists and amateurs will all be the order of this Personal Computing Show. A free copy of PERSONAL COMPUTING magazine will also be given to each attendee.

Computer enthusiasts are encouraged to participate in the show. Since the primary purpose is to explain all aspects of personal computing to the public, there will be a need for dozens of workshops and seminars. Plans call for publishing all papers in a Show Proceedings to be made available after the show is over.

If you are interested in participating in this show, you are asked to contact *David Bunnell* or *Louise Garcia* (505) 266-1173, no later than August 15, 1977.

Personal Computing Magazine
401 Louisiana SE
Suite 'G'
Albuquerque, NM 87108



□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

MONTREAL, NOV 16-18

The Queen Elizabeth Hotel in Montreal, Canada will be the site of a gathering on all aspects of mini- and microcomputers and their applications. Symposia on mini- and microcomputers will be Nov. 16-18 and on personal and home computers Nov. 17-18. A computer show will be held Nov. 17-18.

There will be several awards presented including a microcomputer for the best paper in the symposium on Personal and Home Computers. The judging will be based on both the content and quality of presentation.

A 200-250 word abstract should be submitted by September 1, 1977 to the Symposium Chairman. Notification of acceptance will be sent by September 10. The accepted papers are due by November 10. The proceedings are scheduled to appear February 1, 1978. Accepted papers are considered to have been submitted for possible publication in the ISMM journal Mini- and Microcomputers.

For correspondence, submission of abstract and to be placed on the mailing list:

Prof. J. L. Houle – MIMI '77
Ecole Polytechnique,
Case postale 6079, succursale A
Montreal, Quebec, Canada H3C 3A7
(514) 344-4753

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

OTHER

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

MICRO COURSE IN NY

The Evening Division and Department of Mathematics & Computer Science of ST. JOHN'S UNIVERSITY offers a series of intensive short courses on low-cost personal computing. Each course is designed to provide an understanding and a well rounded body of information on successful implementation and use of small computer systems. It reviews the state of the art in current microcomputer technology including both hardware and software design as well as numerous applications of personal computing in education, recreation, business, etc. No computer expertise is required. Any individual who has a common sense understanding of computers can actively benefit from the course.

The course will meet every Tuesday from 6–8 P.M., from Sept. 27 through Oct. 18, and it costs $20. For further information:

Dean Patrick Basilice
Evening Division
ST. JOHN'S UNIVERSITY
Jamaica, N.Y. 11439
(212) 969-8000, x101

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

ROBOTICS GRANTS

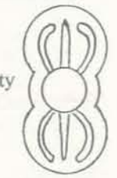
Three individual grants of $100 are offered by the United States Robotics Society to students who survey practical activity in research and development on robots in specified areas of the world. The surveys must be performed for academic credit with formal approval of appropriate professors.

With the sudden rise in the use of personal, privately owned computer systems, private research and development in robotics and artificial intelligence has surged. More than seventy members of USRS[tm] alone report active work on robots. The Society is seeking an estimate of robotic activity worldwide, and these first grants are the beginning of a general search for the robots.

Grants will be made for surveys of: The United States West of the Mississippi, the United States East of the Mississippi, and Canada. Later grants will be made for surveys of other areas.

The reports will be published as part of the basic robotics literature, establishing their authors and supervisors as important contacts in the field. Proposals from applicants are due on or before 30 September 1977; completed reports are due on or before 20 June 1978. For details, write:

Survey Grants
United States Robotics Society
Box 26484
Albuquerque NM 87125



□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

NAME-THE-USERS'-GROUP CONTEST

A five-year membership is the prize in a name-the-users'-group contest sponsored by computer hobbyists exchanging information on the use of Heathkits. The more obvious choices are ruled out by Heath Company objections to use of their trademarks by an independent organization. Entries and requests for further information should be sent to:

Charles A. Floto
267 Willow Street, Apt. 27
New Haven CT 06511

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

MICRO COURSE IN CA

Chaffey Community College at Alta Loma, California, has scheduled a fall-quarter course in microcomputer programming that will be offered in two 12-week class sections. Both classes will be from 7 to 10 p.m., one on Tuesdays starting Sept. 13 and the other on Wednesdays beginning Sept. 14. The only prerequisite for the three-unit course is that a student must be a high school graduate or 18 years of age. There is no tuition fee entailed for state residents.

Donald J. Ketchum
Data Processing Professor
Chaffey Community College
Alta Loma, CA 91701

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

AEDS PROCEEDINGS

Educational data systems are a function of hardware, software and peopleware. The fifteenth annual convention proceedings are now available from the Association for Educational Data Systems. Over eighty original papers presented April 25-29, 1977, in Fort Worth, Texas, are included in the publication. Categories include:

Instructional Support & Curriculum
Managing Computers and Computing
Information Systems & Applications
Certification
Human Values in Educational Data Systems
Social Implications of Computers
Computer Impact on Society

Copies may be ordered by sending $10.00 to:

AEDS Proceedings
1201 16th Street, N.W.
Washington, D.C. 20036

□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□▫□

PERIODICAL GUIDE

The *Periodical Guide for Computerists* is a new 20-page book that indexes over 1,000 personal computing articles from 15 magazines for January - December 1976. The articles are indexed under more than 100 subject categories. Indexed are magazine articles, letters from readers, book reviews and editorials from both hobbyist and professional publications.

The books are available from E. Berg Publications, 1360 S.W. 199th Ct., Aloha, Oregon 97005 for $2.50 each postpaid, and also from local computer stores.

A forthcoming issue in July will index January - June 1977 articles.