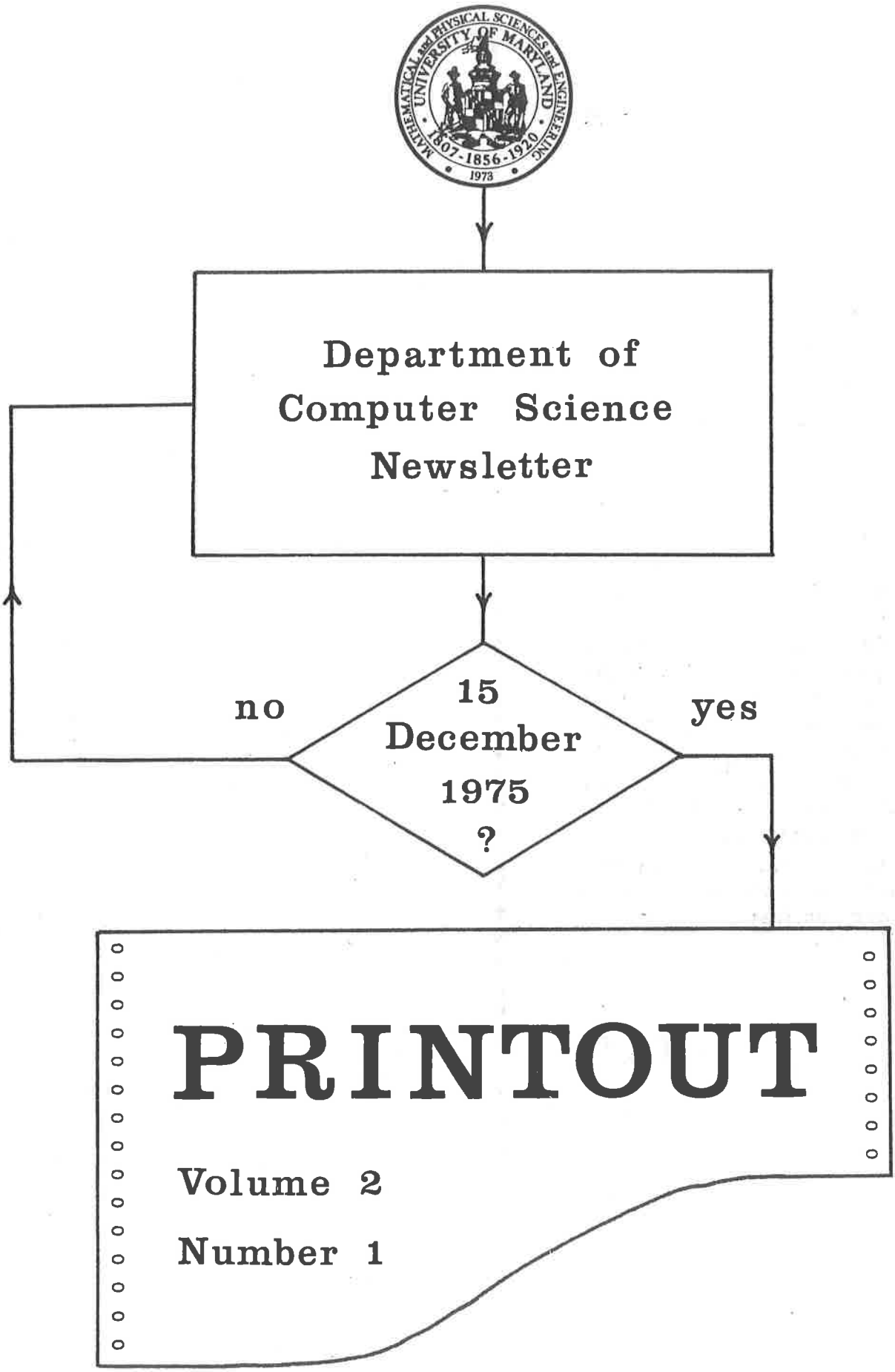

MATHEMATICAL and PHYSICAL SCIENCES and ENGINEERING
UNIVERSITY OF MARYLAND
1807 - 1856 - 1920
1973
Department of
Computer Science
Newsletter
no
15
December
1975
?
yes
PRINTOUT
Volume 2
Number 1

PRINTOUT, the newsletter of the Department of Computer Science of the University of Maryland at College Park, is published sporadically and distributed to faculty, staff, and students in the Department. Opinions expressed in signed articles may be those of the author, but no opinions represent the policy of the Department, or of the College Park Campus, or of the University.

Contributions may be submitted to the editor, and unless they are obscene or seditious they will probably be used, but minor editing may be done. Complaints directed to the newsletter will be investigated and publicized when possible. It is well to keep in mind however that the Department is subordinate to higher levels of administration, not the other way around; and, the Department does not provide computing service to the campus. Complaints in these areas are best directed to other publications.

STAFF

EDITOR

DICK HAMLET

PHOTOGRAPHY

PAT BASILI

TYPING

ELEANOR WATERS

PRINTOUT

Volume 2
Number 1 & 2

CONTENTS

PROGRAMMING LANGUAGE RESEARCH

To describe Maryland's programming language research, the ingredients are faculty members Basili, Zelkowitz, Noonan, Hamlet, and Gannon; the SIMPL family of languages and the PLUM PL/1 system; and the ideas of modeling, design, implementation, evaluation, correctness, and testing. A (slightly misleading) table may then be cast:

PLUM	SIMPL
Zelkowitz: *implementation, evaluation*	Basili: *modeling, design, implementation*
Noonan: *correctness*	
	Hamlet: *implementation, testing*
	Gannon: *implementation, evaluation*

The table is misleading because sharp boundries are hard to draw. For example, although Zelkowitz is primarily concerned with PLUM, he keeps an eye on SIMPL -- where PLUM acquired its CASE. Similarly, although Gannon may actually work with the SIMPL compiler, many ideas originated in his PL/1 background.

A more precise description must be given according to who is doing the research.

V. R. Basili

Professor Basili is interested in modeling programming languages and in designing languages that can be used to implement the next generation of large scale systems.

A primary goal is the development of well-defined application oriented languages. For this reason language modeling has played a major role in his work. He has used a hierarchical graph modeling facility (HGL) for defining the semantics of the languages which he has designed. The model has even been used in systematizing a set of ground rules for high level language design. The process carries the language design from a high level model all the way down to the implementation using a step by step, top-down approach imposing more and more structure on the language components as the design becomes solidified.

His language design efforts include GRAAL, the SIMPL family and SL/1. GRAAL is a graph algorithmic language for describing and implementing graph algorithms of the type arising in applications. The language contains both sets and graphs as prime time data elements and a compiler for a FORTRAN based version of the language is available on our system.

The development of a set of application oriented languages motivated the idea of a family of programming languages and compilers. The family concept implies a set of languages designed as extensions to a base language with the respective compilers built as extensions to the base compiler. The function of this was the SIMPL family of programming languages of which locally available SIMPL-T and SIMPL-R are members. Work is currently underway on the design of SIMPL-D which will include a data definitional facility for implementing data abstractions. A data abstraction is a data type or structure along with the appropriate operators and access mechanisms, respectively. The facility permits the user to gear the language to the particular problem.

Other language efforts include SL/1, a language under development for the CDC STAR 100. It was designed to bridge the gap between the users' needs and the efficient use of the specialized vectorized machine.

Dr. Basili has also been involved in the development of the technique of iterative enhancement for program development. The technique involves the stepwise development of the software system beginning with a simple initial implementation of a properly chosen (skeletal) subproject which is followed by the gradual enhancement of successive implementations in order to build the full implementation. Quantitative measures for the technique have been developed to examine the data relationships of the successive iterations.

Current research includes the design and modeling of language features to be added to the various languages in the SIMPL family, the development of measures for evaluating these features and the investigation of software development techniques and measures.

M. V. Zelkowitz

A different approach toward programming language design is being taken by Professor Zelkowitz. Under the assumption that languages like FORTRAN or PL/1 will be around for a long time to come, what help can a compiler be in writing proper programs. Pl/1 has been called a "Fatal disease"; however, subsets of PL/1 are very powerful languages to use. Towards this end, the PLUM compiler has been implemented. PLUM is not PL/1, but a "super-subset" of PL/1. That is, the better features of PL/1 have been maintained, other less desirable features have been deleted, and other extensions to the language have been added. For example, a CASE statement has been added.

The compiler also checks for proper structures. Various messages are produced warning the user of correct PL/1, but poor usage. Such things as superfluous blocks, lack of comments, possible structure or parameter errors, while correct Pl/1, are extremely error prone and are detected by the compiler and passed on to the programmer for evaluation.

Current research is in evaluating the program development process. Over the summer, a data collection facility was added to PLUM. Each program writes 84 words of information to a mass storage file. This data includes compile time, execute time, number of statements, and errors, size of program, number of blocks, etc.

This data will be analyzed over time in order to evaluate the programming process. Problems to be investigated include: do proper comments decrease development time? Does demand usage decrease development time? Does goto usage increase development time? The current work is basically a large data management problem. The large amount of saved data must be efficiently retrieved and analyzed. Currently, PL/1 programs are being written to access this data.

This last statement brings up an interesting problem. PLUM was designed to compile efficently, but not generate the most optimal code. However, there doesn't seem to be any other language for the 1108 which has powerful computational facilities, an easy to use control and data structuring facility, and efficient file management facilities. If any readers know of anything, please contact Dr. Zelkowitz.

R. E. Noonan

Professor Noonan's primary focus has been in the area of proving correctness of computer programs. At the current time only small, simple programs can be proved correct. Practically speaking, this means programs having under 400 lines of code and involving only integer arithmetic and simple variables and arrays as data structures.

The major impact of research in correctness techniques has been that we now better understand programming as a methodology. For example, structured programming was developed as part of the research effort into proving correctness. Dr. Noonan's research currently involves

trying to extend structured programming from a mostly syntactic methodology to one involving the semantics of what the program is doing. One offshoot of this is the beginning of a theory of good comment placement and content. It may be possible in a year or two to modify both the SIMPL and PLUM compilers to analyze comments and issue suitable warning messages.

Another aspect of his research has been to try to extend correctness techniques to large programs. Such programs are normally designed using block levels of abstraction. Dr. Noonan's research has been to develop formal specification techniques for such programs. Unlike HIPO diagrams this allows "what if" type questions to be posed and answered (via proofs) before any coding has begun. Such techniques may help us develop a structured design methodology for large programs.

R. G. Hamlet

Professor Hamlet was trained as a systems programmer in ALGOL 60 on the Burroughs B5500, so he is an advocate of high-level systems implementation languages. The departmental laboratory's PDP-11 systems, and the self-compiling SIMPL-T compiler, offered the opportunity to try this idea on a minicomputer, and the result (with a good deal of help from Dr. Zelkowitz and a compiler-writing class) was SIMPL-XI, a cross-compiler which generates absolute PDP-11 code and provides enough features to make assembly language unnecessary. (The system which links the laboratory PDP-11s to the 1106 is written entirely in SIMPL-XI, for example.) Building on the base of a much improved SIMPL for the NOVA minicomputer, Dr. Hamlet is now rewriting SIMPL-XI, partly to eliminate the snarl created by too many independent programmers who developed it, and partly to allow experimentation with ideas involving absolute minicomputer code which retains the advantages of relocatable modules and run-time debug packages, without the overhead.

TURN TO PAGE 16...

Professional News

JEFFREY WEN-HU YEH received the Samuel Alexander Fellowship Award from the Washington chapter of the ACM in June. LARRY DAVIS was awarded honorable mention.

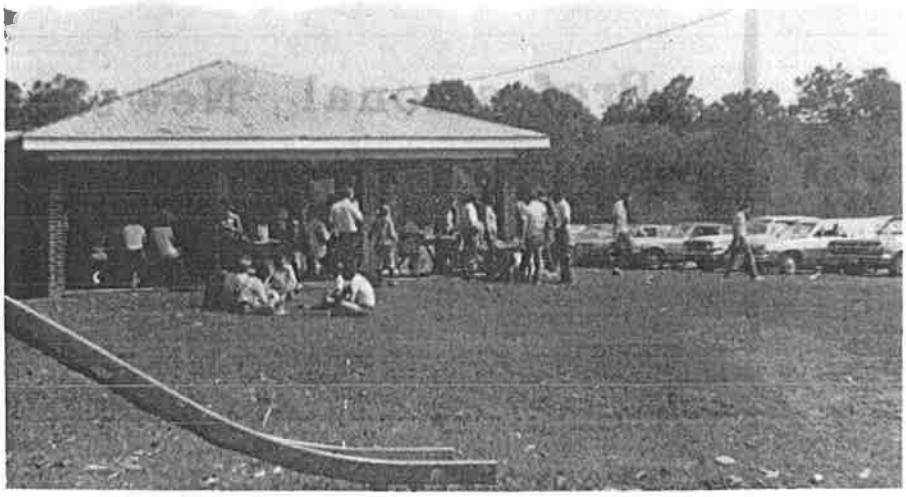
Professor AZRIEL ROSENFELD lectured at the Hungarian Academy of Sciences, Budapest, as part of the US-Hungarian Joint Seminar on Pattern Recognition in June. He also spoke to the Japan Industrial and Technical Association, Tokyo, in August, as a part of a seminar on pattern recognition.

Language design, and the graph language GRAAL, were the subjects of several talks by Dr. VICTOR BASILI in June and July. He spoke at the Technische Hochschule Daimstadt, Daimstadt; at the Institut fur Informatik, Stuttgart; to the Gesellschaft fur Grafische DV und Stukturerkennung, Schloss Birlenghoven; and was a principle speaker at the Workshop Graphen-Sprachen und Algorithmen anf Graphen at the Technische Universitat Berlin.

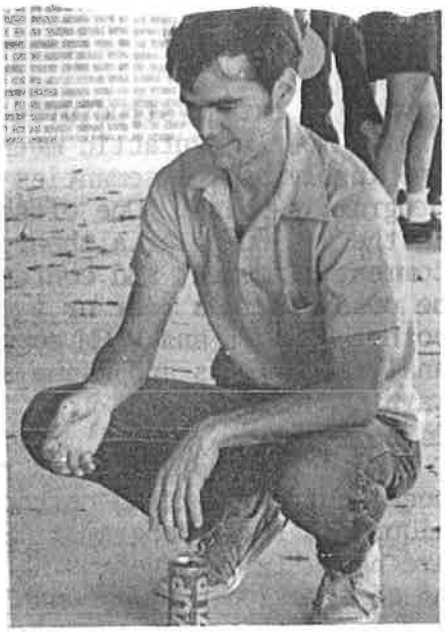
Dr. HARLAN MILLS is now a part-time Visiting Professor. Mills and Professors JACK MINKER and WERNER RHEINBOLDT have been appointed to the Advisory Editorial Board of the NBS/NSF Software Engineering Handbook.

Professor JACK MINKER presented lectures at the NSF Workshop on Advanced Automation at Purdue University in October, and at the Fourth Texas Conference on Computer Science at the University of Texas, Austin, in November. He also participated in the ACM/NBS Workshop on Data Base Systems held in Fort Lauderdale in October. Minker continue as Awards Chairman of the Washington chapter of the ACM, and has recently been appointed ACM/NBS representative to coordinate activities between these groups.

Dr. STANISLAW BUDKOWSKI of Warsaw is visiting the Department this year, working with Professor Chu.



DOES ANYONE REMEMBER THE PICNIC?



Yoga, anyone?

Turnaround: 5 minutes

Say cheese!

Music hath charms...

"And then he said..."

Mingling

"Everyone was there, dear."

The Ol' Mud Hole

- Pat Basili

Publications, Etc.

Anderson, N.S., S.M. Pine, and A. Rosenfeld. Derived scales for degree of simultaneous contrast in six Benussi ring figures. Bull. Psychonomic Soc. 6. 1975. 289-292.

Basili, V.R., R.E. Noonan, and M.V. Zelkowitz. A programming language curriculum in computer science. IFIP 2nd World Conf. on Computer Education, Sept. 2-5, 1975.

Basili, V.R., and A. Turner. A transportable extendable compiler. Software Practice & Experiences, Vol. 5, No. 3, July-Sept. 1975. 297-278.

Basili, V.R., and A. Turner. Iterative enhancement: a practical technique for software development. Proceedings of the 1st National Conf. on Software Engineering, Sept. 1975.

Basili, V.R., and Terry Baker. Structured programming. Institute of Electrical & Electronics Engineers, Inc., IEEE Catalog No. 75CH1049-6.

Cranston, B., and R. Thomas. A simplified recombination scheme for the Fibonacci buddy system. Comm. ACM 18, (1975), 331-332.

Davis, L.S., and A. Rosenfeld. Detection of step edges in noisy one-dimensional data. ibid., October, 1975, 988-991.

Fishman, D., and J. Minker. π-representation: a clause representation for parallel search. Artificial Intelligence 6 (1975), 103-127.

Hamlet, R.G. On execution traces, with an application to understanding programs. Univ. of Md. TR-421, 1975.

Hamlet, R.G. Compiler-based systematic testing. Univ. of Md. TR-423, 1975.

Ibarra, O.H., and C. Kim. Fast approximation algorithms for the knapsack and a sum of subset problems. J. ACM 22 (1975), 463-468.

McSkimin, J., and J. Minker. The use of user supplied semantic information in resolution-based QA systems. IEEE Conf. on Theorem Proving, June, 1975.

Minker, J., G. Wilson, and S. Bennet. An algorithm for π-answer extraction. IEEE Conf. on Theorem Proving, June, 1975.

Minker, J. Association memories and processors: a description and appraisal. Encyclopedia of Computer Science & Technology (J. Belzer, A. G. Holtzman, and A. Kenst (eds.)) Marcel Dekker, Inc., N.Y., (1975), 283-324.

Noonan, R. Structured programming and formal specification. IEEE Natl. Conf. on Software Engineering, Sept. 11-12, 1975.

Noonan, R. Towards a canonical form for computer programs. ACM Natl. Conf., Oct. 20-22, 1975.

Rosenfeld, A. Fuzzy graphs. In L. A. Zadeh et al., eds., Fuzzy Sets and Their Application to Cognitive and Decision Processes, Academic Press, N.Y., 1975, 77-95.

Rosenfeld, A. Networks of automata: some applications. IEEE Trans. SMC-5, May 1975, 380-383.

Rosenfeld, A., L.S. Davis, and J.S. Weszka. Region extraction by averaging and thresholding. ibid., 383-388.

Rosenfeld, A., and E. G. Johnston. Digital detection of pits, peaks, ridges, and ravines. ibid., July, 472-480.

Rosenfeld, A., and J. S. Weszka. A comparative study of texture measures for terrain classification. Proc. Conf. on Computer Graphics, Pattern Recognition, and Data Structures, (IEEE Publ. 75CHO-981-1C), May 1975, 62-64.

Rosenfeld, A., S. W. Zucker, and L.S. Davis. Picture segmentation by texture discrimination. Proc. 2nd U.S.-Japan Computer Conf., August 1975, 14-17.

Rosenfeld, A. Picture processing: 1974. Computer Graphics and Image Processing 4, June 1975, 133-155.

Rosenfeld, A. Sequential and parallel automata. In E.R. Caianiello, ed., New Concepts and Technologies in Parallel Information Processing, Noordhoff, Leyden, 1975, 123-135.

Rosenfeld, A., and E. G. Johnston. Low-cost interactive image processing. Proc. Soc. for Info. Display 16, 1975, 1-7.

Rosenfeld, A., and J. S. Weszka. An improved method of angle detection on digital curves. IEEE Trans. on Computers C-24, Sept. 1975, 940-941.

Rosenfeld, A. A note on automata detection of texture gradients. ibid., (Oct.) 988-991.

Wilson, G., and J. Minker. Resolution, refinement, and search strategies: a comparative study. IEEE Conf. on Theorem Proving, June, 1975.

Wilson, G. An approach to parallel inference and parallel search. IEEE Conf. on Theorem Proving, June, 1975.

Zelkowitz, M.V. An integrated software development and evaluation tool. Univ. of Md. TR-395, July, 1975.

Jucker, S.W., A. Rosenfeld, and L.S. Davis. General-purpose models: expectations about the unexpected. Proc. 4th Intl. Joint Conf. on Artificial Intelligence, Tbilisi, USSR, Sept. 1975, 716-721.

functions(dummy variable)

A hostel is an element of a set of process dwellings. A Distributed Computer Network (DCN) is a group closed under the operation of hostel attachment. Hostel attachment. Hostel attachment is associative, commutative, and distributes over the entire virtual memory. A hostel has an inverse, called a capability, which is used to call the supervisor. The supervisor contains a set of re-entrant homomorphisms called the kernel and is protected by inspired hardware against unauthorized access, including disinterrment by squirrels.

A message is an uninterpreted sequence of syllables which has a nonpositive entropy. A DCN forms a ring under division by messages. It follows immediately that messages go in circles with ever-decreasing information until disappearing down a black hole. Thus, a message is a function which maps the domain of a hostel into a virtual space with a nonzero protect key. The inverse of a message is called a retransmission request.

A segment is an element of virtual memory which obeys Asimov's first law. Other elements of virtual memory are called nonresident and pay foreign income taxes. Segments can be swapped or bartered freely, but a tax must be paid for each channel command. The revenue collected in this manner pays the rent on the hostel.

A process is a two-dimensional matrix, the elements of which are called ports and used to load and unload messages. Processes are elements of hostels which have the capability to grok segments. Occasionally too many segments are grokked in this manner and some processes become address exceptions. Under these conditions the system goes into thermal runaway and some processes may melt. An elegant solution is to disolve a message or two in odd parity turpentine until a process boils off into another hostel. However, the resultant broth contains bits of floating point which not only can infect the program status word, but smells like hell.

Most of our knowledge of hostel systems comes from a Dr. Rebraf, a veterinarian from New York, who discovered that processes have the ability to permeate protection rings at cryogenic temperatures. Messages generated by processes under these conditions circulate forever or until an entry point is found at less than ten kilogauss in the logout region.

Dr. Rebraf went on to explore the field formed by the addition of processes modulo-two. If a shared file is used to smooth a process, the result is another process called a demon. Demons occur naturally at certain atomic numbers, and can be made to fission by segment bombardment. At energies above a megabyte demons become nonresident and cannot be found for days. During such times the printer must be shut down or its lid flies open and knocks the coffee cup all over the read-only memory, which then turns write at location zero and sails out to core on a floppy disk.

May the bird of paradise redundancy all over your floor.

-- D. L. Mills

SEQUENCES

What is the next number in the sequence

10, 11, 0, 1, 110, 111, 100, 101, 11010, 11011?

Turn to page 18 when you give up.

Is there a next element in the sequence

Fishman, Milgram, Cook, Yeh, ... ?

Carriage Control

There seem to be four excuses for a newsletter, and this issue of ours is so late that I've had rather too much time to mull them over. (Mull means to stir with a hot poker.) The primary function of publishing a house organ (you'll have to look that one up yourself) is communication. People who work together only see each other five days a week, so the newsletter is to make sure they know something about each other. Secondly, a newsletter serves as a reflexive back-patting device. By publishing all the good, impressive things about what is going on, and none of the bad, lackluster things, we seek to improve others' impression of us, and our own self-assessment. But the trouble with patting yourself on the back is that it may lead to a broken arm. Thirdly, PRINTOUT seeks to be informative. For examle, in our first year, we suggested that CMSC 620 was our worst graduate core course, just after its content had been changed to correct this. We explained all about setting up 1108/1106 class accounts so that the computers are even more overloaded than they had been.

We are left with the fourth function of a newsletter, which like the fourth estate is sometimes not there when it sould be: humor. I approve of technical jokes. The more specialized the vocabulary, the more surprising it is when words are set on their heads--surprise seems to be the heart of humor. But as Walt Kelly was fond of pointing out, the primary function of humor is to puncture the overblown and pompous. Nowhere are the balloons so ripe for the pin as in professional, technical subjects. A profession might be defined as an occupation which has existed for a long enough time to accumulate a critical mass of self-righteousness. This is codified as the ethics of the profession, providing euphuisms for bad conduct. For example, instead of calling someone a bigot, you need only say that he is "lacking in professional courtesy." Our most deservedly former President carried the idea to its logical conclusion when he tried to hide his misdeeds under the one-man profession of "presidency." And of course, a professional is a person who needs or enjoys the protection his occupation provides. If you think this kind of humor is worthwhile, just try it out on some doctor, lawyer, or scientist. The best response you are liable to get is, "What?" which means, "Who are you to attack the respectable institution to which I am devoting my life?" Ah, well...

-Dick Hamlet

Pamela Smith (nee Zave)

Pamela Z. Smith received an A.B. degree in English from Cornell University in 1970, after writing an honors thesis on the late 18th century novel of manners. Having been seduced by the IBM System/360 Programmer's Reference Manual, she did graduate work in computer science at the University of Wisconsin. Her current research interests include deterministic process modeling, asynchronous communication mechanisms, state graph abstractions of parallel processes, and hierarchical computation structures. She intends to publish under her maiden name of Zave.

Her favorite recreation is wilderness backpacking, during which she has discovered a special affinity with moose. "We both like to stand with our feet in the water." Favorites among the activities she would pursue in Maryland if she ever got more than a week ahead of her students are landscape watercolors, guitar, and tennis.

WHAT KINDA TEST D'YA LIKE?

Two examinations are given below, representing extremes in the uncertain territory of teaching and learning. All of the answers are on page 17.

1. PROGRAMMING APTITUDE TEST

This is part of a short test designed by the Massachusetts Educational Data Processing Association.

1. If you went to bed at 8:00 PM and set the alarm to get up at 9:00 AM, how many hours sleep would this permit you to have?

2. Do they have a 4th of July in England?

3. Why can't a man living in Winston-Salem, N.C., be buried west of the Mississippi River?

4. How many birthdays does the average man have?

5. If you had only one match and entered a room in which there was a kerosene lamp, an oil burner, and a wood-burning stove, which would you light first?

6. Some months have 30 days, some have 31 days, how many have 28 days?

7. If a doctor gave you 3 pills and told you to take one every half hour, how long would they last you?

8. How far can a dog run into the woods?

9. A farmer had 17 sheep. All but 9 died. How many did he have left?

10. Take two apples from three apples and what do you have?

11. An archaeologist claimed that he found some gold coins dated 46 B.C. Do you think he did?

12. A woman gives a beggar 50 cents. The woman is the beggar's sister, but the beggar is not the woman's brother. How come?

13. Is it legal in Massachusetts for a man to marry his widow's sister?

14. You are completely blindfolded. Placed in front of you is a box of stockings, all the same size. 25 of the stockings are white; 25 are black. What is the minimum number of stockings that you can remove and be abosolutely sure of a matching pair?

15. What four words appear on every U. S. coin besides "In God We Trust"?

16. You have a baseball. You throw it away from you as hard as you can. It does not hit anything, nor does anyone catch it, but it comes back to you. There are no strings or elastics involved. Why does the ball come back?

17. Are there more doorknobs on the right side of doors or on the left side?

18. You have 4¼, 5½, and 3-2/8 haystacks and you put them all together. How many haystacks do you then have?

19. A small planeload of Americans flying over Canadian territory meet with a fatal accident. None of the remains can be identified. In which country will the survivors be buried?

2. MACHINE-SCORABLE TEST

A. True-false. Mark each item T or F. Do not explain your answer even if you can.

1. A bit in a binary computer is larger than a bit in a trinary computer.

2. A context-free grammer means the same thing no matter who is talking about it.

3. A PDP-11 was made after a PDP-10, and a PDP-6 before a PDP-10, but a PDP-8 was made after a PDP-6.

4. Exec VIII is better than Exec II.

5. The answer to this question is "F".

B. Fill-in-the-blank. Write exactly one word in each blank. Do not explain.

1. The digital computer is _____.

2. When a program cannot be proved correct, it is probably because it contains _____.

3. Exec VIII error messages are _____.

4. If a _____ is _____, then it is unlikely to _____.

5. "@@ERROR - _____."

C. Multiple-choice. Mark the single best answer. Do not explain.

1. The octal code for ASCII STQ is:

   a. 001
   b. 002
   c. 005
   d. 023
   e. None of the above.

2. The mnomonic for the Univac 1108 "skip if less than" instruction is:

   a. SLT
   b. TL
   c. TLS
   d. SL
   e. None of the above.

3. Which of the following is not a computer?

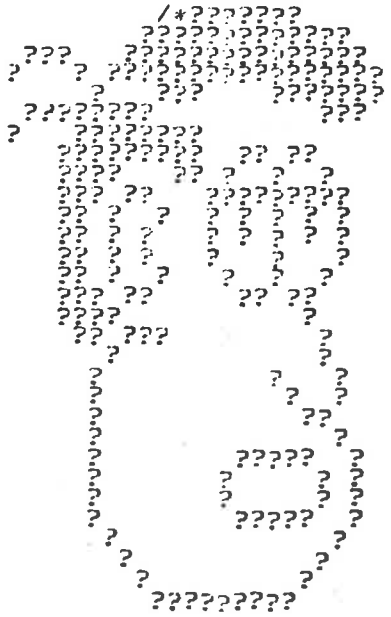
   a. IBM 701
   b. Burroughs 205
   c. Philco 202
   d. GE 225
   e. None of the above.

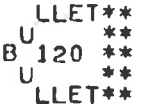
4. Which of the following is not a FORTRAN variable?

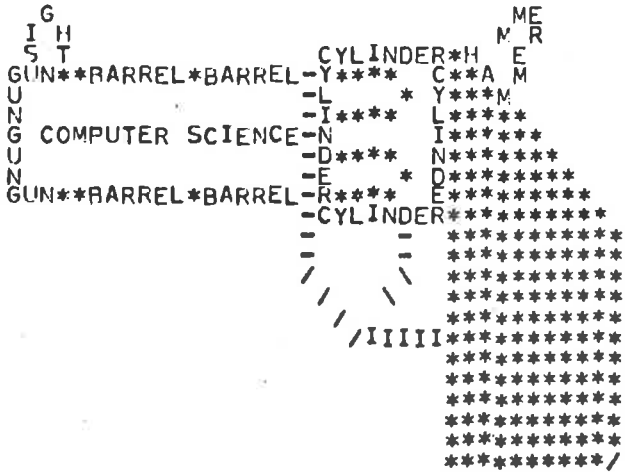
   a. A
   b. AA2
   c. B B B
   d. A$B
   e. None of the above.

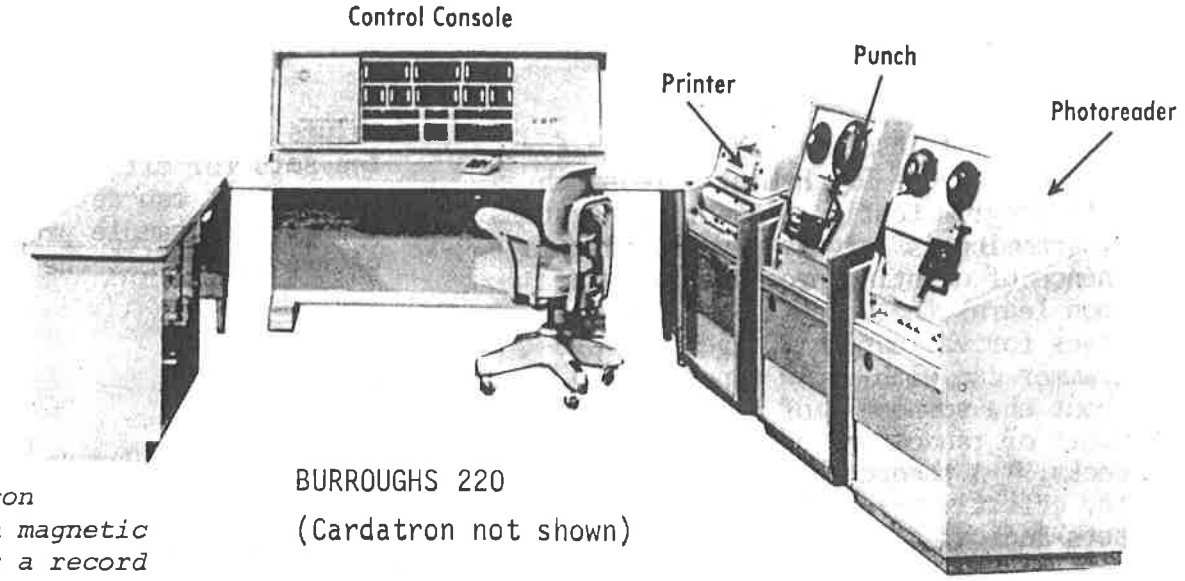
5. The answer to this question is:

   a. a
   b. a or b
   c. b but not a or c
   d. if a then d
   e. All of the above.









BURROUGHS 220
(Cardatron not shown)

6. *Burroughs acquired Electro-data Corp. for their interface system to IBM peripherals, but the hardware was less than reliable.*

7. *The Cardatron system used a magnetic drum to buffer a record before feeding it to rD. Programming equivalent to a keypunch drum card could be effected with a format band written beside the data.*

In the 220 there are lots of cores
For the decimal words--they're arranged in fours[8].
In those four bits you can put 15,
And if you do, it breaks the machine.
If numbers that large had been in God's plans
We'd have three fingers more on each of our hands,
But the Burroughs 220 doesn't give a damn.

Dawn is here, you've done all you can,
When into the room comes a little old man.
With his Simpson meter[9] and thermos of black,
He starts poking at the memory rack.
In a minute or two he says, "Son of a gun!"
And then you know you can junk your run,
But the Burroughs 220 doesn't give a damn.

If you have trouble let me tell you what to do:
There's people in the Center, there to help you.
There's a guy named George who can make it go
As soon as he gets back from Mexico.
You get straight answers, they never shirk,
Straight from the shoulder: "It doesn't work."
But the Burroughs 220 doesn't give a damn.

As this recital comes to an end,
Here's a bit of advice: If you need a friend,
You'd better look near and far away,
But watch out for friends that are green and gray.
A digital computer seems mighty shrewd,
But most of the time you wind up screwed,
But the Burroughs 220 doesn't give a damn.

-Dick Hamlet

8. *The 220 had up to 10,000 words of 11 digits.*

9. *I always thought this man was reading the power supply voltages. But one summer the 220 was down for two full weeks while almost every circuit was replaced: the power supply regulator had failed, the supply had drifted 30% off its specs, and ruined most of the machine. Maybe the man was just trying out his meter.*

PROGRAMMING LANGUAGE RESEARCH
FROM PAGE 5

SIMPL-T itself is the vehicle for another part of Dr. Hamlet's research, into program testing. Practical systems programming is largely devoted to mainanence of existing programs, and one soon learns the value of concise test cases for validating changes. A programmer can with a few inputs learn more about the soundness of his patch than hours of random use of standard "test decks." A theoretical investigation of the effectiveness of finite test data sets indicates that there are two interesting kinds: (1) data which forces the program through every program path which exists, and (2) data which justifies the complication of every program expression in that no simpler expression would do. (In the sense (2), value 0 is a bad test case for a variable X in the expression X*(Y + 1), since the less complicated X or even just 0 would do as well in that case.) Unfortunately, the theoretical situation is that although these interesting sets exist for every program, there is no general algorithm for finding either kind.

It is intuitively correct that no system can devise "good" tests for a program--only the programmer can do so. But a system can very easily check tests that the programmer supplies. In the implementation, which uses the SIMPL-T compiler in a novel way to perform tests using compiled code, yet retain contact with the source program names, the programmer supplies specifications in the form

/+ TESTS (x_1,y_1) ... (x_n,y_n) +/

where the pairs represent input/output values. The compiler then not only checks the sets for efficacy in the above two senses, but can detect incorrect output. The trivial sample program at the bottom of the page shows the system working.

J. D. Gannon

Professor Gannon suggest a quiz:

1. What is the value of 5 - 4 + 3?

2. The PL/I repetition statement shown below is executed twice.

```
do  I = 0, I = 1;
    .
    .
    .
end
```

What is the value of I on the first repetition? on the second?

If your answer to the first question was 4, you were probably drawing on your experience with arithmetic in reading from left to right. However, if you are familiar with a well-known conversational programming language whose adherents are addicted to "one-liners", you might read the expression right to left and answer -2. In question two, I has the value zero both times the repetition statement is executed. The expression I = 1 yields the value false ('0'B in PL/I) which is assigned to the "integer" variable I. Obviously we are suffering from confusion between the symbols for assignment (the first =) and comparison for

```
        INT FACTOR=5
        INT FUNC TRIVIAL(INT PARAM)   /+TESTS (0,0) (3,15) +/
1         /+TESTS (0,0) (3,15) +/
2         CASE PARAM OF
3           \0\  RETURN(1)     \1\  RETURN(0)
5           ELSE RETURN(FACTOR*PARAM) END
          START

TESTING MAIN ROUTINE 'TRIVIAL'...
 ON INPUT 0, THE OUTPUT OF 1 SHOULD BE 0.  CONTINUE?
YES
 STATEMENTS NEVER EXECUTED:
  3
 GLOBAL VARIABLES WHICH DO NOT VARY:
  FACTOR
```

equality (the second =) aggravated by the implicit conversion of logical values to integers.

Designing language features which are easy to translate, but difficult for programmers to use, makes it difficult for people to develop programs that have a high probability of being correct. Appropriate language design can both reduce the probability of making errors and increase the probability of detecting those that are made.

Dr. Gannon approaches the problem of language design from a "human engineering" viewpoint. Language features are analyzed to determine what errors result from misunderstandings about the features (i.e., characteristic errors). Hypotheses are constructed about the frequency and severity of the characteristic errors of alternate language features which perform similar functions. Experiments involve groups of subjects solving problems with programming languages. Each group uses a language which differs only in the feature being studied. All runs by the subjects are collected and analyzed for the characteristic errors of interest.

Bibliography

V.R. Basili. A structured approach to language design. J. of Computer Lang. 1 (1975), 255-273.

V.R. Basili. The SIMPL family of compilers. TR-304, 1974.

A.J. Turner & V.R. Basili. Iterative enhancement: a practical approach towards software development. Natl. Conf. of Software Engineering, Washington, 1975.

M.V. Zelkowitz. An integrated translation and evaluation tool. TR-395, 1975.

R.G. Hamlet. Other people's monitors. SIGPLAN Notices 8 (1973), 21-22.

R.G. Hamlet. Compiler-based systematic testing. TR-423, 1975.

J.D. Gannon. & J.J. Horning. The impact of language design on the production of reliable software. Int. Conf. on Reliable Software, Los Angeles, 1975. (SIGPLAN Notices 10 (1975), 10-22.

Answers to the tests on page 11. Are you peeking?

1. Programming Aptitude Test

Evaluation of Responses

Number Right	Interpretation
19	You cheated.
16-18	Aptitude of the highest caliber. You could be president of IBM.
13-15	Excellent--capabile of managing a full unit-record installation.
9-12	Good--possibilities as a systems analyst or computer circuit designer.
5-8	Have you thought about collecting unemployment benefits?
3-5	Menial tasks or teaching duties.
0-2	Would probably make a good administrator.

Answers.

1. One (no AM or PM on the alarm clock)
2. Yes (and a 5th and a 6th etc.)
3. He is still alive.
4. One.
5. The match.
6. All of them.
7. One hour. (one at 8, one at 8:30, one at 9)
8. Halfway (after that it is running out of the woods)
9. 9.
10. Two apples.
11. No (how would the minter know it was 46 years before the birth of Christ).
12. The beggar is the woman's sister.
13. Hardly (if it's his widow's sister, he must be dead).
14. Three (only possible combinations are 3W, 2W/1B, 1W/2B, 3B, each combination guarantees a pair).
15. United States of America.
16. You threw it straight up.
17. Same numb er on each side.
18. One.
19. Bury the survivors?

2. Machine-scorable Test

A. 1. F 2. T 3. T 4. T 5. -

B. 1. down 2. statements 3. incomprehensible 4. program--nontrivial--work 5. SYNTAX

C. 1. e 2. e 3. e 4. e 5. e

Ann Arbor, 5 Nov.

Dave Mills provided the November 5, 1975 issue of the University of Michigan Computer Center Newsletter, because it contained the following news item:

HERMES and Multiple GRABs

HERMES, MERIT's terminal support system, now supports multiple GRABs. The user may open as many as four connections simultaneously from a single terminal. To distinguish between the connections when FLIPping, the FLIP device command now takes several different types of parameters. The connections are numbered 1, 2, 3, 4 in the order in which they are acquired; e.g., FLIP 2 FLIPs to the second connection GRABbed. Each connection also has a name, in the format xSnn, displayed when it is GRABbed and each time it is FLIPped to; e.g., FLIP LS04 FLIPs to that particular connection. Or the user can FLIP to a particular host: FLIP MS or FLIP UM, for example. (If more than one connection is open at the specified host, HERMES FLIPs to the first one it finds.) Finally, the user can simply FLIP forward or backward through the open connections, one at a time, with FLIP + or FLIP −. (FLIP alone is equivalent to FLIP −.)

He threatens to lock anyone who can figure it out without assistance in the PDP-11 laboratory for twenty minutes, and/or explain what it means.

There are two other interesting things in this issue of the Michigan Newsletter, obviously not unusual there:

(1) Some impending changes to the Michigan systems are announced, which are expected to occur on December 17, and January 5, respectively.

(2) Almost half the Newsletter is devoted to some of the suggestions recieved from computer users, and informed, useful answers to them. Some of the answers are of the form "that's too hard to change, or not a good idea because..." while others are of the form "that has (or will) be done."

Although the Michigan Newsletter is a nice professional job, it is evident that the real professionals at Michigan are in the systems and user support organizations.

(Unless you are obeying a GOTO on page 9, turn back to that page.)

The next number is $11000_{-2} = 8_{10}$; the sequence is arithmetic, starting at -2.

Negative bases have been unjustly overlooked by hardware designers (perhaps because they have a positive number of fingers and toes). There are certain advantages to negative bases for internal computer use. For examples, there is only one representation for zero (a claim that signed absolute value and one's complement can't make), and a sign bit is no longer needed since both positive and negative integers can be represented as unsigned strings over {0,1}. Of course, negative bases do have a drawback or two. For example, base negative two adders require two carry bits. Perhaps the most unusual aspect of negative-base numbers is that the ability to represent some integer x in n bits does not imply the ability to represent $-x$ in n bits. In the usual case for computers where n is even, the largest possible positive integer representable is:

$$2^{n-2}+2^{n-4}+ \ldots +1 = 1/3(2^n-1) < 2^{n-1}-1$$

and the smallest possible negative integer is:

$$-(2^{n-1}+2^{n-3}+ \ldots +2) =$$
$$-2/3(2^n-1) < -(2^{n-1}-1)$$

Anyone intrigued by unusual number bases should enjoy a paper by D. Knuth (CACM 3, 245) on the bizarre base $\sqrt{-2}$, in which all positive, negative, and complex numbers with integral parts can be represented by strings over {0,1}.

-R. M. Lefler

From *Malice in Blunderland*, Thomas Martin:

COMMITTEE: (1) A collection of the unfit chosen from the unwilling by the incompetent to do the unnecessary.

(2) A group of people who, individually, can do nothing, but collectively can meet and decide that nothing can be done.

EDUCATIONALLY SPEAKING

The purpose of this column is to keep you informed of what is going on in the General Committee on Educational Affairs. This committee (more popularly known as the Education Committee) concerns itself with courses, enrollment, admissions requirements, etc.

The committee is made up of the entire faculty, including instructors, as well as two graduate students (Franklin, and Reiter) and two undergraduate students (Bryl and Wilder). As such, it is the most broadly based committee of the Department. Occasionally this means that the Committee will consider matters not within its jurisdiction simply because the right people are in attendance.

The Committee normally meets about once a month. This is possible because its subcommittees do most of the real work. For example, the scheduling of courses is handled by the Field Committees: Computer Systems, Information Processing, Programming Languages, Theory of Computing, and Numerical Analysis. Additionally, ad hoc subcommittees are appointed as the need arises. All of these committees have student representation.

The rest of this column is devoted to itemizing the major business either transacted or in progress.

Increased Enrollment

At the August 29th meeting Dr. Austing reported that our enrollment figures are up versus last year. Over 1900 students are enrolled in our courses. Enrollment is up in 103 and 110 but down in 120. Graduate enrollments seem to be continuing their decline of recent years. Weighted credit hours (which purports to be a measure of faculty workload) appear to be 15-20% higher than last.

Committee to Review CMSC 110/120

A committee has been appointed (chaired by Dr. J. Vandergraft) to review the course content of CMSC 110/120. The committee is to consider outlines for each course and recommend texts. Additionally, the programming languages to be covered are to be specified. Finally, the background needed to skip CMSC 110 must be specified. The student representative is Mr. J. Bryl.

Committee to Review M.S. Degree

Dr. R. Hamlet was chosen to head a committee to review the M.S. degree program. Based on enrollment statistics and M.S. degree statistics, it is hoped to develop a quantitative picture of the difficulty of obtaining an M.S. degree. The committee is to recommend changes in degree requirements as needed. Specifically, the committee is to determine the feasability and need for offering a late afternoon/early evening curriculum leading to an M.S. degree within 2 to 3 years.

Meeting with IFSM Faculty

On November 14 the Education Committee met with the faculty of the Information and Systems Management group. One immediate consequence of these contacts is that we are represented at their meetings and Dr. T. Hardgrave is their representative at ours. Additionally, they requested that we consider if and how to initiate 3 undergraduate CMSC courses for IFSM use (and perhaps other departments as well). Finally, we have been invited to meet with them (across campus) sometime during the Spring semester.

Change in Appeal Procedure

At its last meeting the Education Committee voted to change the appeal procedure for the (graduate) Comprehensive Exam. From now on a single appeals committee will be appointed to hear all appeals in a given semester. The student must appear before the committee, at which time the committee may examine (orally) the student in the areas in which he contests the results of the comprehensive exam. The committee must still report its recommendations to the professorial faculty.

-R. E. Noonan

Department of
Computer Science
Newsletter

15
December
1975
?

yes

no

PRINTOUT

Volume 2

Number 2