THE TENTH RAND COMPUTING SYMPOSIUM

Fred Geunberger, Editor

P - 3998

PREFACE

This Paper is an expurgated and condensed transcript of the Tenth Annual Computer Symposium held at The RAND Corporation, 13 November 1967. These one-day sessions on topics in computing are held on the day prior to the Fall Joint Computer Conference--a juxtaposition that has made it easy for top men in the information-processing field to contribute their time and talents. The transcript, which has been edited from the original by each of the attendees, reflects serious but unprepared thoughts on the topic, "The Teaching of Computing." The views expressed are those of the attendees and not necessarily those of their employers or of The RAND Corporation.

SUMMARY

The Tenth Annual RAND Computer Symposium held in Santa Monica, 13 November 1967, dealt with the single topic, "The Teaching of Computing." RAND was represented by George Armerding, and Consultant Fred Gruenberger, who organized and chaired the session. Other attendees were:

Richard Andree, University of Oklahoma
William Atchison, University of Maryland
George Cannon, San Fernando Valley State College
Charles Davidson, University of Wisconsin
David Evans, University of Utah
Aaron Finerman, State University of New York
Bernard Galler, University of Michigan
Don Krehbiel, Santa Monica City College
Roger Mills, TRW Systems
Norman Sanders, The Boeing Company
Melvin Shader, IBM
Joseph Weizenbaum, Massachusetts Institute of Technology
Robert White, Informatics, Inc.

Directly or indirectly, these men have collectively been responsible for the computer education of a significant number of all the people in the United States who have had formal academic exposure to information processing.

During the seven-hour session, the attendees discussed primarily the people to whom computing should be taught; the grade level at which computer education should begin; the training of programmers and those who will teach computing; and the advice that should be given to high schools, junior colleges, and trade schools to help them initiate computer education programs.

The original transcript was edited and corrected by each attendee. This Paper is the final version of the transcript.

THE TENTH RAND COMPUTING SYMPOSIUM

Fred Gruenberger, Editor

The RAND Corporation, Santa Monica, California

GRUENBERGER: Messrs. Krehbiel, Mills, White, and myself, as you may know, were involved last spring in a teaching project which resulted in a film. We don't want to ram this film down your throats, but I know that most of you have not seen it, and it is pertinent to the theme of this discussion. If you concur, we will show it after lunch.

DAVIDSON: I would like to raise a point of information. Each of the topics on the agenda uses the word "computing." Must we limit our discussion to the subject of training programmers, or are we open to other aspects of the information-processing field?

FINERMAN: I have always understood that the term "computing" was the more comprehensive term and includes other aspects of information processing as subtopics.

GRUENBERGER: Well, let's back off a little, Charlie. I'm sure that you all appreciate that we do not intend to restrict discussion in any way and that people will wander off on any topic they prefer, no matter what restrictions we lay down.

There is an aside here. Bernie just suggested that maybe we should look at the movie this morning instead of this afternoon. Is there any feeling for that?

WEIZENBAUM: I have a feeling that if we see the film first, we will be discussing the film rather than the agenda topics.

MILLS: I've seen the film, and I would vote that we have it later on.

KREHBIEL: I would suggest, since my interest is in computing as a broad topic, that the first item on our agenda will fire off the discussion quite nicely and will permit us to take up things in their proper chronological order. I think we should logically consider where computing begins and where it goes from there, so I suggest that we take up the agenda topics as they now stand.

GRUENBERGER: Fine, we'll start off with the number one agenda topic, "To Whom Should Computing be Taught?"

FINERMAN: It seems to me that one way to approach this topic would be to consider to whom computing should not be taught. Perhaps we may find that there are relatively few people in the latter group.

GALLER: What do we include by "computing"?

SANDERS: The distinction is between what computers can do and how to program them.

GRUENBERGER: Didn't you want to give a speech on this subject anyway, Charlie?

DAVIDSON: Yes, I'd like to refer to the Pierce report, which lists three levels of understanding for which one might strive. Although this report is concerned primarily with college-level work, its pertinence is much broader. The first group that they discuss is the professionals; namely, the computing-science majors who intend to make some aspect of computing their life's work.

The second group might be called the tool-user class: scientists, engineers, linguists, and sociologists, who are

probably not interested in the computer *per se* but only in what it can do for them.

The third group--and by far the largest--needs to understand the uses of computers for computer appreciation. They need to be able to interact with computers in their daily life. They need to be informed sufficiently to be able to vote intelligently about the use of computers in the school systems, to understand the construction of data banks, and even to read the popular press intelligently. In our discussion I would like to make sure that we are including all three of these groups.

This report was made for the President's Science Advisory Panel by a group headed by John Pierce of the Bell Telephone Laboratories.

GRUENBERGER: It would seem reasonable that we agree now that whenever we speak of computing we have in mind all three groups so that we don't have to qualify our statements continuously.

MILLS: I'll agree except for one part of the statement that Davidson made which implied, I thought, that the professionals would be the principal users of computers.

DAVIDSON: I didn't mean to imply that.

SHADER: What's wrong with terms like "information processing" or "information sciences"? I don't want to get hung up on words, but it seems to me those terms are broader. If we used the broader terms it would include uses of computers in social sciences and the humanities, and thus extend our discussion to all possible uses.

DAVIDSON: I would agree. Those groups use computers, and that's why we are interested in them.

CANNON: Moreover, communications facilities are used quite extensively. "Information Sciences" is more descriptive of our subject matter.

WHITE: Within our own field we seem to be in agreement on what is meant by information science, but people in other fields--the telephone company, for example--seem to have an entirely different meaning for that term.

DAVIDSON: I think we can discuss quite intelligently with our own understanding of the terms, whereas if we were to issue a press release, we might have to phrase it quite carefully.

FINERMAN: No group in our discipline has been able to agree on a name; I doubt whether this group will be more successful.

DAVIDSON: Or should be.

SANDERS: Or needs to.

KREHBIEL: The content of our discussion will be conditioned by which group you are referring to. One group, for example, has need for information processing in a much broader sense than we usually mean by the term computing. I'm wondering if we have suddenly narrowed our view just to the college population, or are we going to include also high school graduates, high school students, and drop outs.

DAVIDSON: I would certainly not want to restrict the discussion. I simply took off from the Pierce report, which deals only with the college level. I do think that the subdivision into three groups given in this report applies to any level including high school students.

CANNON: That was the original question: to what group should computing be taught?

WHITE: I doubt that we have to worry much about the first group mentioned by Davidson; namely, the professionals in computing science. They will be the least of our worries today.

GALLER: At least that group is best taken care of so far.

MILLS: Because of the wording of the question, the thing that will hang us up is the word "taught." That third class of people mentioned in the Pierce report--the largest of the three groups--must be informed, but not necessarily taught. They need to know, for example, that when a negative pay check is issued, it wasn't a computer that goofed but that some program was written badly. When these people are told that a programmer goofed, they are being informed, not taught in the normally accepted sense.

You're not teaching so much what computers do in this case as what they don't do. It's certainly part of our job to disseminate this sort of information as widely as possible. Whether it comes under the heading "computing," I don't know.

WHITE: For the greatest benefits to the profession in the long run, I think that we should concentrate our attention on the third group; that is, the group that needs courses in computer appreciation. A course in computer appreciation should not be a single course taught at the standard of college level work, but should be spread out over the entire school curriculum. I guess that means the real problem now is the training of teachers.

DAVIDSON: That has been our experience. Once you have the general faculty involved, computer appreciation spreads quite rapidly.

ANDREE: What would you give these teachers?

WHITE: As I understand what Charlie is doing, the content of his computer appreciation course is exactly what I would recommend to all teachers now in training. That content should be required everywhere.

DAVIDSON: Let me come back to a point that I think slipped away from us. If you want these people to understand the cause of the negative paycheck, then it is my conviction that the most powerful way to do this is to subject them to the experience of trying to communicate with a computer.

With all due respect to Dick Hamming and his night school lectures, I don't think you can teach understanding of computing without having access to a computer. Hamming maintains that he is looking ahead one or more decades to when we will have to indoctrinate many hundreds of thousands of people, and it will not be feasible to tie each of them to an actual computer.

Hamming's viewpoint may be correct for a time that is somewhat distant, but now I maintain that intimate access with a machine is vital to understanding--to get them to understand the mentality, the meticulousness, and the stupidity of computers.

FINERMAN: I assume that you are talking about the communication involved in writing programs. You're not talking about teaching the actual operation of the machine.

DAVIDSON: That's correct.

SHADER: Then it seems to me that we should go back to the question that was raised earlier; namely, is there anyone to whom computing should not be taught?

ANDREE: I want to second that idea. I find it difficult to imagine anyone to whom we purport to give an education who should *not* be subjected to some kind of course, be it Hamming's, Davidson's, or anyone else's that might be called computer appreciation.

GRUENBERGER: I'll offer a couple of examples: specifically, my wife and my son. I maintain there is a large group of people to whom you can't teach anything about computing in any way, shape, or manner.

ANDREE: I have spent several hours talking with your wife, and she already has more understanding of computers than you're going to get at the end of most of these courses.

GRUENBERGER: Maybe so, as the result of osmosis over a number of years.

MILLS: I would argue that anyone can be taught an understanding of computing if they can read. We are, of course, excluding the mentally handicapped. I maintain that anyone can be taught computing, and I would take exception to Charlie's stand in that I think it can be done without having access to a computer. Admittedly, it can be done much better with that access. With the machine you get a better response, a deeper understanding, and a faster progression. I think it would be a shame for us to neglect teaching a group of five hundred people simply because at the moment we cannot hook them all up to a computer. We have been educating people for years in various subjects without access to the thing we were describing.

SANDERS: Something seems a bit illogical here. We have in the world cyclotrons and radio telescopes, and we don't educate everyone about those, do we?

CANNON: But there's a difference! This great mass of people will learn much about computers whether we do anything about it or not. If nothing else, people gain some education about computer systems when they are cautioned not to fold, mutilate, staple, or spindle. I find it difficult to imagine any literate or educated person these days not having picked up some knowledge of computers. If we assume they are going to learn something, then it becomes our duty to worry about what is taught, who teaches it, and how it is taught. It is essential that the general public's view of computerized systems be in true perspective, that what they learn be correct, and that the attitudes formed be sound.

SANDERS: People will learn about the results regardless of what happens to that card. Does it really matter that they know, as long as the card is not mutilated?

CANNON: I think people will have to interact with the system in ways that make it highly desirable that they understand something about what is going on.

FINERMAN: Is there any device in the world, whether of recent or long duration, for which everyone has been educated?

GALLER: How about the telephone?

FINERMAN: Well, wait a minute. You misunderstand me. People generally don't know anything at all about the workings of a telephone, but only how to pick it up and dial it.

DAVIDSON: But at least they've learned how to communicate with it.

FINERMAN: Well, that's my point. Three quarters of the people in this country who drive an automobile, myself included, have no knowledge whatsoever of its internal workings. It seems to me that the computer appreciation course that we've been discussing, which includes communicating with the machine through programs, goes far beyond the level of education that we require of people for these other devices. I can teach someone how to communicate using a telephone without having a telephone around. Also, I think I can teach someone how to communicate with a computer without having a computer around.

SHADER: I wonder if we could use that as an operational definition; namely, to teach someone how to use it.

WEIZENBAUM: Two analogies come to my mind at the mention of the automobile.

Let's consider the knowledge that people have of psychology. I'm thinking now of the knowledge of psychology that people have who can be considered educated whether or not they have had a college education. I go back in my mind to the time of Sigmund Freud, around 1910 to 1920, at which time knowlege of the subject was very much restricted to a

small group of people. Today we can see the effects of teaching psychology in the way people discuss subjects like permissiveness in their children and in the way they worry about whether they are traumatized or not. The point is that today there is a great deal of popular knowledge on this subject (even admitting that some of it may be wrong). Nevertheless, the bulk of the population in this country has some appreciation of the subject matter without in any sense being experts at it.

Most people have never seen a psychologist at work but still know most of the basic principles. Most people who have been through college have a rather good knowledge of the subject. The knowledge may not be correct, but there is an appreciation of the subject. I think we must look forward to the time when that same situation will exist with computers. Although it may not be for another ten years, it probably is not too far from today.

GRUENBERGER: But I think we would all find it appalling if the knowledge that people had about computers, acquired in that same manner, were as bad as what I'm sure the psychologists believe the public knowledge of their subject is.

WEIZENBAUM: Let me go then to the analogy about the automobile. I do think that a great many people in the United States have a pretty good knowledge of how an automobile operates.

Most of you probably know that there is a chess-playing program now in existence at M.I.T., and it is pretty good. It will beat anyone who walks in off the street. The initial version of this program was written by two twenty-year-old youngsters. They wrote it in two weeks for the PDP-6, and today when they are dealing with a much fancier version, it still occupies only 6000 words of core. It makes you wonder how this can come about. There have been years and years of work by Ulam, Bernstein, and others.

These kids came along and in two weeks wrote a program that was in many respects far superior. Art Samuel and I were discussing this program a few weeks back and kept repeating this same question, how does it come about that these kids can do this? I don't want to go into a detailed explanation of how this is possible, although I do think I have one.

I think part of the explanation lies in the following point. If you imagine a medical doctor in India asking a fourteen-year-old boy, "How long do you think it would take me to teach you to operate my automobile?" the boy would probably look at the calendar. If you picture the same situation with a fourteen-year-old boy in Iowa, on a farm (if you can imagine finding a fourteen-year-old farm boy in Iowa who does not already know how to operate a car), he would probably look at his watch. The difference in learning time is several orders of magnitude.

The difference is that the automobile is an intimate part of our culture, and people assume all sorts of things about it. I think that in a relatively short time, say a few years, in the advanced countries like the United States this same situation will exist with computers. Today it is highly specialized. The two kids who wrote the chess program grew up in the atmosphere of M.I.T. Tomorrow it will extend to a great many more kids.

FINERMAN: Let me expand on my earlier remarks. I don't want to argue what is desirable or what is undesirable at the moment; I think it is obviously highly desirable that high school kids going into college be exposed to computing. However, we do have, both at the college and secondary school level, the question of priority as to what should be taught. We have urgent questions today concerning the overall content of the undergraduate curriculum, whether it is possible for the undergraduate to specialize at all or

whether he should be exposed to a broad range of fundamental ideas. This matter is being discussed more and more these days, and in these discussions the teaching of computing is assuming some prominence. Regardless of our feeling about computing, we must worry about what takes precedence over what.

I am sure that high school kids can be exposed to computing technology in such a manner so that they know it is around. At a minimum they should learn that if they devote more time to the subject at a later day, they can learn to communicate with the computer and use it. They should certainly be exposed to examples of what a computer does. However, all this can be done rather casually. At the high school stage I don't think students need a language course, a programming course, access to a computer, or assigned problems to be solved.

WEIZENBAUM: I'm sorry, I didn't think we were talking about that level of teaching. I thought that for the moment we were talking about the general public. I was not talking about formal courses in any way; hence my analogy to the spread of knowledge about psychology. Most people who go through high school today never get any explicit teaching of the concepts of psychology; nevertheless it is in our culture. I am suggesting that, no matter what we do, in the next ten or twenty years the kind of pervasive attitudinal knowledge that people have about psychology and automobiles will, in fact, be present for better or worse in our culture. I come to that conclusion before I even begin to discuss the subject of teaching.

Another aspect of this is that today teachers in the elementary schools are begining to talk about computing. For example, the teacher of my thirteen-year-old daughter told her that if you ask a computer to divide by zero, smoke will come out of the machine.

GALLER: We could arrange it.

WEIZENBAUM: I interpreted this comment by my daughter as a question, and it provided me an excellent opportunity to discuss with her the whole concept of division by zero. It seems to me that the presence of computer knowledge in our culture will offer many such pedagogical opportunities.

The point is that the teacher in this case did enunciate a mistaken idea, a misleading idea, and, I think, a harmful idea both with respect to mathematics and computing. But teachers will do this. They will market ideas to children, whether right or wrong, and this is one of the ways that computing is entering our culture now.

FINERMAN: I don't know that there is anything you can do about teaching misleading ideas since mathematical concepts such as division by zero have been around much longer than computers. Even in the absence of computers, if you went to many teachers in the elementary schools and brought up the topic of division by zero, you would get somewhat the same reaction; namely, that smoke will come out of anything, even the paper you're writing on. It seems to me that simply exposing these teachers to computing in a casual manner will not help that much, since they have had deeper exposure to mathematical concepts without much help.

WEIZENBAUM: But you've turned my example upside down. I was trying to illustrate something about computers and not about mathematics. I was trying to show that our culture has now reached the point where arguments are being made to children in computer terms. It's independent of what the argument is about. It is conceivable that a teacher of civics twenty years ago would have framed his arguments in terms of something like the Gallup Poll; this teacher might now be putting his arguments in computer terms.

GRUENBERGER: Now hold on just a minute. I would guess that thirty years ago there must have been a group

like this sitting around a big table arguing about the relative merits of loading up the high school curriculum and the undergraduate curriculum in college with courses in Spanish. As the result of that sort of discussion, I spent five hideous semesters trying to learn about Spanish, and I didn't learn it. After five semesters I managed to pass a formalized attainment exam so that they would let me out of that bind, but to this day I know exactly one sentence in Spanish and that one I memorized cold, the same way I can memorize the digits of pi.

SANDERS: But you learned something about English in the process.

GRUENBERGER: No, I didn't. I'm sorry, Norman. I would say in retrospect that the one thing I learned was to hate the idea that someone can conclude that everyone should learn Spanish in high school and in the freshman year in college. That is the sum total of what I learned. English is a separate subject, and if someone is supposed to learn English (and I might add that I am all in favor of that), then the thing to do is to study English, not study Spanish in order to learn English.

I'm simply objecting to this flat-out statement that says there is no one to whom we should not teach computing. I submit that there is a large group of people (those who are artistically inclined, for example) to whom this is an unbearable chore. They will reap no benefit whatsoever, and we should not put ourselves in the position of flatly stating that everyone should be subjected to some course in computing.

I'm really surprised that such a stand should be taken here. The people in this room were not selected at random. They represent collectively most of the people who have taught most of the computing students over a long period of time (and from a basis of knowledge) as opposed to those

who are now teaching computing on the basis of six months of experience with the field. It would seem clear to me that it must have occurred to those who have done this teaching that every semester you have in one or more of your classes, undoubtedly by mistake, someone who should not be there and who cannot learn the subject no matter how hard you try and, in fact, no matter how hard he tries. There is a certain level of abstract symbolism and mathematical notation that is basic to the learning of computing and that is utterly beyond some people.

Now I am not saying that that's particularly bad. I think this group of unteachables is probably pretty small. I'll just say it again; I object to flatly stating that everyone should be exposed to a computer course.

FINERMAN: As I understand what Weizenbaum said, the thing I was agreeing with was not the teaching of computing but the pervasiveness of computing as an all-powerful force in our society. People will be exposed to it and will speak in terms of computing regardless of whether or not there are formal courses for them. It is an environmental factor.

GRUENBERGER: I couldn't agree more; after all I am a computer man. But the intent of our discussion related to whether or not there was anyone who should not be exposed by force to a course in computing, and I still think in terms of people I know, like my son, for whom such a course is a waste of time because they will learn nothing from it. In fact, they will fight it.

MILLS: I object to the concept of forced exposure to a course. In the first place, I don't think anyone said that we had to have a course. I think, Fred, you are the only one who has put it in terms of a formal course. I realize that you are now in the teaching business.

Education is not entirely formal courses. For example, if we could even educate the press so that they would quote

something now and then halfway right, we would have half the battle won. And if we could educate the teachers, we would have the other half of the battle whipped.

I object to the concept of formal courses. At almost every college there are formal courses in the slide rule, and if you happen to miss them by transferring between schools, you sure learn the slide rule in a hurry when you transfer into thermodynamics. Similarly, we should have courses in college in computing, but I don't think that we should force people to take them. We are talking about educating the public, and that is part of our job.

WHITE: But if you are going to do that, there is a class of people who should be forced into such courses--those who are going into teaching. Regardless of the level they are going to teach, it seems to me that they are obligated to learn something about computers.

ANDREE: I don't think that you have to force them to take a formal course. At Oklahoma we have many "courses" that are simply one-session, two-session, five-session, or ten-session lectures, and they are mobbed. We have frequently had to ask the press not to publicize one of these offerings and to restrict it to people on campus because we had a room that would hold only so many people. These are sessions that carry no credit; people simply come if they want to know. The sessions are always filled.

GALLER: I would like to raise a question. Supposing at this meeting and meetings like it we decided to do nothing. I think we are agreed that in some amount of time computing and its concepts will permeate our society just as the concepts of psychology have done. The question I have is, if we do nothing and similar groups do nothing, will the end result be good or bad? We should consider in what way they will be good or bad and then figure out what we should do about it.

MILLS: We can look at the situation right now since computing has been around for at least ten years.

DAVIDSON: Are there not some basic concepts that should be forced onto the general population by whatever means is appropriate? It might be through formal courses, casual lectures, or the indoctrination of the press, but is it not true that there are some basic misunderstandings current in the population that should be cleared up, and is it not our duty to do so? It strikes me that much of the confusion in the popular mind about computers stems from one or two basic concepts. One of these, for example, is the notion that a computer has no judgment but does only what it is told. If we could spread just that idea around, it would help to counter the false gospel that is being spread.

WHITE: And yet I keep thinking of the chess-playing program that Joe brought up; that is pretty hard to explain in those terms.

WEIZENBAUM: I agree that there are some basic concepts, but I think it would be harmful if we succeed in spreading the notion that the computer does only what it is told. The concept that is referred to requires deep understanding. I think that it is dangerous to have the concept in the public mind that the computer does only what it is intended to do.

A concept that I would regard as much more useful and that I think should enter the public metaphor--the imagery that the public uses to think about all sorts of things, not only computers--is the idea that might be stated as "building big ones out of little ones." This is the notion of higher hierarchical modules, relating not only to hardware (that is, how the computer is constructed from the gadget viewpoint) but from the point of view of programming. The notion of a subroutine, for example, is such a basic concept. We, as computer people, are aware of how crucial the concept of the subroutine is, but it turns out that this concept is also crucial in life.

DAVIDSON: I'd like to point out that there is a difference between what I said ("the computer does only what it is told to do") and what Joe just said ("the computer does only what it is intended to do").

WHITE: That's a description of an automatic transmission; as a matter of fact, it is a subroutine in the car.

WEIZENBAUM: That's exactly correct. Another such concept is the input-output idea.

The one I like best though is the idea of making big ones out of little ones. This is the concept in which you take a whole lot of small, disparate things, put them together in a systematic way, and somehow obtain a whole that is greater than the sum of its parts.

I think basic concepts like these can be understood by the general public on many different levels and thereby enter the public metaphor.

DAVIDSON: But why?

FINERMAN: What purpose does it serve?

CANNON: I'd like to support the previous comment suggesting that it is somewhat dangerous to foster the view that the computer does only what it is told to do. Fred Gruenberger had an article in *Datamation* some months back on this very point, in which he showed that we provide the instructions to the machine, and then we are not capable of saying what the machine will do with the very instructions that we wrote. This is true only because we work so slowly and unreliably that we do not have the time to fully and correctly consider the entire logical path that the computer will follow as it traverses the branches and iterations of our program. The individual who is not knowledgeable about computers will not understand the distinction.

There is a dilemma here in that the computer does only what we tell it to do, and at the same time it can beat us at chess and do other things that in practice we can't predict. It seems to me that the public must be made to understand how we get from here to there. If they don't understand this transition, they will build up in their minds...

EVANS: I don't know how to resolve that particular dilemma, but I think that what should be done is related to Joe's earlier remark. The computer is a force that, in all probability, will alter our lives more than the automobile did. It is a new thing of a different kind. Because it has been around for 10 years and people regard that as a long time is the reason we are here.

We talked of the analogy to the automobile. In the early days of the automobile those users who were not automobile builders had to have a fairly detailed knowledge of its workings in order to use it. That was the situation 10 years after automobiles were invented. Here we are now, 10 years after computers have become widespread, and already we have reached the point at which it is impossible for most users to have an understanding of how the device works. Not only have we shortened the time span with computers, but we may have bypassed the stage in which a fairly large number of people understand the inner workings. The analogy isn't perfect, since the automobile is a very special-purpose device and it is pretty obvious what it is intended to do. It is not at all clear what a computer is intended to do nor do I think we are ready to state in any way what the sociological implications of the computer are. As it turns out, in retrospect we were not able to predict the sociological implications of the automobile either, but I think the implications of the computer will be even more broad.

The marriage of information-processing technology and communication technology will more quickly change the society in which they are embedded than did the automobile. When we talk of education in relation to the computer, we should be prepared to arm people with the knowledge they need to appreciate the possibilities. Thinking people need to be guided so that they can control this social revolution. It may be that the best way to arm people is to teach them to communicate with the machine--I don't know. It's a different kind of thing than the automobile; you don't take it on; it takes you on.

ATCHISON: I would certainly agree very strongly that education in computing science is urgently needed. I feel that both aspects that we have been talking about are important. A large amount of computing knowledge will be interpolated into our society and into our educational system. I, too, have noticed the effect on my fourth grade child of computing knowledge disseminated in the classroom, most of which is relatively correct. Our kids are getting it all the way up and down the line.

About two weeks ago a group of us met at Boulder, Colorado, to discuss the implications of the computer in the secondary school curriculum. The discussion ranged down to the junior high school level, and the idea that came out, which I don't think we have dwelt on here yet, is that it furnishes a good way to plan. Computing technology furnishes a way of orienting oneself and a way to approach things. The concept of flowcharting, for example, is an important concept to get across to youngsters. If we can, in addition, get a little of the computer into the discussion, it would provide a strong motivating force. The basic concept will help us to understand many other concepts such as the checkless society. Computer appreciation courses may be one answer, but I think that computing

knowledge should be interpolated into many courses, such as basic science courses in our high schools.

ARMERDING: What happens if we don't teach computing appreciation? Can we imagine a group like this discussing the subject of automobile appreciation fifty years ago? Do you suppose that the people who produced automobiles at that time, looking ahead to the widespread use of the car by the public, would have said to each other that courses in automobile appreciation would be necessary so that the public could intelligently use the beast? I seriously doubt whether anything like that took place. You can argue that perhaps they should have; that perhaps some of the ills brought upon us by the automobile could have been avoided that way. But the fact remains that the automobile did spread itself by osmosis in a natural way. Even Finerman understands how his automobile works, though he says he doesn't.

Look at another aspect of it. If we devote a substantial share of our resources to the teaching of computer appreciation, we may be taking away resources that we need for something I think is far more important, namely, the teaching of computer professionals. What good will it do to teach computer appreciation to the general public when the fifty thousand programmers that we need (or whatever the number is) are not being properly taught?

SANDERS: In the early days of the automobile, according to an old movie I saw the other day, there were people who went around smashing them up because they were going to be a curse to society. Are we expressing a fear that somehow the general public will revolt against the computer?

GALLER: There might be a small danger there, but I don't think it is justification for the sort of thing we are talking about.

The automobile is an interesting example. It seems to me the effect of the automobile on our society has nothing to do with whether or not people were trained how to drive them or take care of them. The changes in our society produced by the ease of transportation represented by the automobile would probably have come about regardless of people's knowledge of the workings of the automobile. The effects are in a quite different area. If we had had organized discussions years ago of the social implications of the automobile, we might still be in a bind today about problems of pollution, traffic control, freeway networks, etc. The real social effects of the automobile have come upon us unexpectedly.

I like the thought of teaching large numbers of people how to program, and I think large numbers of people should learn this; but I am not so sure that everybody should. I think, as with the automobile, the real effects of computers are going to be in other areas. Given limited resources (as we always have), I think we can obtain the best leverage on the problems being discussed by a system of distributed logic, such as aiming at the training of teachers. If we could get the message to the teachers and the press, they could in turn distribute it (rightly or wrongly) to the general public.

KREHBIEL: We seem to be concentrating on just one group (namely, the general public) who will be influenced by computers but will not be users as such. I'd like to focus attention, at least briefly, on the users. My thinking has been greatly influenced as a result of reading Andree's book, and I am wondering whether we do need to teach computing. In the first few pages of that book you introduced your readers to the idea of reading GOTRAN, which is an abbreviated form of FORTRAN. The book makes it relatively easy to introduce students to the concept of translating a problem solution from a flowchart into GOTRAN

language. If a student can produce a flowchart, then he can write the code and use the 1620 computer to solve his problem. He can do this with a very short period of instruction if he understands the concept of problem solving. We are not discussing the problem of problem solving, but we are discussing computing, and in the sense that I am raising, computing is automatic if problem solving is understood. What will happen in five years when professional programmers (which is the other group we have been avoiding) make it easy for people to solve problems without any formalized training in computing? When that happens, the large group that we have been discussing will be automatically taken care of by osmosis.

FINERMAN: I think Krehbiel has raised a fundamental point--that people tend to equate the subject of using computers with courses in communicating with computers through some programming language. It seems to me that the least significant part of communicating with computers is the language. This is true both for people who need only minimal cultural aspects of computing, and those who are going to be users in the sense of problem solvers. We should differentiate between what a computer is used for, using the computer, and solving problems with it. We tend to teach courses in FORTRAN or some other higher level language and assume that thereby students will learn how to use the computer. This is a fundamental misconception. For example, if I write a letter to a colleague in a foreign country in German, French, or Spanish, I can make just as big a fool of myself in the content of that letter as I could if I were writing in English. If my ideas are foolish, my letter will be foolish even if I know the language perfectly. The idea that we should be promulgating is logic, of laying out the plan, whether it be in a formalized problem or in situations of everyday life. This, it seems to me, should be one of the fundamental ideas that we convey.

GRUENBERGER: Weizenbaum expressed that same idea last year. He talked about someone with fluent ability in French trying to write a book on nuclear physics in French.

DAVIDSON: At the Panel on Computer Appreciation held at the ACM Conference in Washington in August, Arthur Kahn of Westinghouse made what I thought was a pertinent point. He made a very effective analogy when he discussed the use of language in the understanding of a culture. You can study the geography and history of a country or of a people, but you don't understand their culture until you can communicate in their own language. Only in that way can you get an insight into their politics, philosophy, and mores.

It seems to me that this idea is essential in a course in computer appreciation. I don't mean by that that people must learn enough of the language to become expert problem solvers. With reference to that broad third class of people, this is not an essential goal, but it is certainly highly desirable, and one of the best ways of achieving the goal is to have them, at least a couple of times, go through the language, examine its restrictions, and see what capabilities are on the other end of the communication channel.

Let me also support a statement that Atchison made a few minutes ago. A formal course in computer appreciation is by no means the only way to go. It is just one of several approaches that need to be taken. I would hope that we don't come to a unanimous decision on a way to go.

WHITE: There is no danger of that.

CANNON: It strikes me that we reached a conclusion, as in the analogies with psychology and the automobile, that everyone will unavoidably gain some form of computer appreciation, for better or for worse. Clearly, then, our duty is to foster information dissemination and formal education that will help the entire public appreciate the computer in true perspective and with desirable attitudes.

Since it is hardly science fiction to conclude that the whole populace will have to interact with computerized systems, the public should be taught the mechanics of how to do this safely, effectively, and happily.

The superficial mechanics of dealing with automobiles were learned quickly and informally. Those who didn't learn stood a good chance of being killed and thus removed from the mainstream of worldly events. No doubt the carnage which does take place on our highways could be reduced by more intensive driver education. It was also suggested that many of our urban problems brought about by the automobile would be less acute today if a more disciplined attempt could have been made to anticipate the possible social and economic ramifications and implications. We would have benefited from sound advance planning. Properly structured education might have fostered better understanding, more constructive attitudes, and sounder habits.

As to attitudes and mechanical competence, the analogy between the automobile and the computer seems realistic to me. For survival, the public will perforce learn how to cope with computers. But the carnage will be less if there is some formal preparation. Furthermore, sufficient understanding is good insurance against the danger of an aroused citizenry panicking in fear of possible social implications. We must not risk having the baby thrown out with the bathwater.

I would also like to comment on a corollary advantage to the general teaching of computing. Computerized systems encourage people to carefully structure the procedure of their activities. Generally, people resent having to do this at first, but many benefits can accrue which extend well beyond the mechanical requirements of the system. After the habits are formed, they appreciate how much can be accomplished with serenity and with a minimum of the turmoil and confusion generated by concentration on putting out fires. Creative thinking flourishes best in such an

orderly environment. Interaction with the computer is an excellent way to learn this lesson. Computing should not be taught without emphasizing the benefits of well-planned procedures.

For that matter, all teachers, across the whole spectrum of education, should convey the idea that a happy balance must be sought between precisely structured procedure and unplanned, uncoordinated freedom. Somehow education has to marry these two objectives. At the moment our educational institutions encourage independent thinking, which is as it should be, but this has to be in balance with the need for carrying out necessary procedures. This is an essential concept, which computing teaches admirably. You adopt a procedure, carry it out, and stick with it religiously until you choose a better procedure. You don't gallop off in all directions at once. If computing taught nothing else, it would be valuable for that alone. We must convey to people that it is to their advantage to carry out sensible procedures with care (not just in computing, but in life in general). Admittedly, there is some danger in this philosophy which may result in too much conformity and lack of initiative. Thus we need militant rebels who will insist that procedures be changed when desirable. But they should be changed as of a discrete date with plenty of advance notice.

GRUENBERGER: Is it our consensus, however, that there is no one left out--that everyone should get some form of computer appreciation?

GALLER: Let me throw in a personal anecdote that may illustrate another aspect of this. We had a family debate revolving around the fact that A didn't want B to tell C that D had had a heart attack. As you can imagine, this was a highly personal thing with personalities and emotions running rampant, but I think the whole discussion changed

its tone when I made the observation that I was unhappy with the manipulation of information that was going on. It made the people in the discussion realize that information was indeed a commodity and that it was being manipulated. From then on the discussion could proceed at that level. I think this is one of the things that an understanding of computing can do for people.

WHITE: I'd like to counter the argument that was brought up in the analogy with the automobile that stated that if you just let it go, it will happen. A basic difference exists between the automobile and the computer. From the time the automobile was first designed and put together it has been a highly marketable commodity. The attitudes toward it and the characteristics of it were largely defined by advertising people. Our approach to the automobile has been conditioned by sales and promotion. We haven't done that with computers, and I doubt that we will in the near future. If anyone is to shape the attitudes and characteristics relating to the computer, it will have to be the people on the inside and not the people on the outside.

WEIZENBAUM: I would disagree strongly with that. It took a very long time for the automobile to show its social effects. When the effects did appear, they were due not so much to the automobile but to the concomitant construction of highway networks, gasoline stations, motels, shopping centers and all that. The people concerned with the automobile industry, up to say 1914, could not have predicted these effects unless they had been much smarter than they turned out to be. A knowledge of automobile engineering wouldn't have helped at all--they would have had to be sociologists more than anything else. It's the feedback loop between these other effects and the automobile manufacturers that engendered the advertising you talk about. I suggest that we, as computer engineers (if you will let

me use that phrase for the moment) are in the same position now as the automobile engineers were at that time. The only way we can predict the social impact of computers is to put our knowledge of computing in the background--not forget it but put it in the background. Our viewpoint should be that of sociologists or psychologists.

When a cultural anthropologist tries to describe what a society is all about, one of his most effective ways is to describe the metaphors that the society or culture uses. What is their system of analogies; how do they describe things? I think it is already upon us (and it will progressively become more pervasive) that the metaphor with the highest currency in the United States will soon become the computer. The computer and all that surrounds it will become, in effect, the road map by which people will navigate through their society. It is terribly important, then, that this road map bear some resemblance to reality. It is terribly harmful if it doesn't. This is where the responsibility lies.

MILLS: A remark was made awhile back about "committing our resources." The resources that we have include the time and talents of everyone in the business. It is everyone's responsibility to do the job that is needed on the broad population that we've been talking about. I agree that we will need professional programmers. I'd like to get some discussion going about that second group of people--the users. Many professional programmers are now in the category of users. The bulk of them have had three years' experience with FORTRAN and are "experts," but they don't have any idea of what is happening to them or what is going to happen to them. Nevertheless, they are busy solving problems--not well and probably not the right problems. If you want to talk about a job of training and education, gentlemen, attack that one! You can always tell a FORTRAN Programmer but you can't tell him much.

SANDERS: I want to return to Joe's reference to social metaphors in which he said that the primary social metaphor of the near future will be the computer. If I understood what he said, it is important that the metaphors be realistic. The social metaphors in ancient Greece, Rome, and Norway were mythology, and that was completely unrealistic. Did this do any harm?

WEIZENBAUM: But it wasn't unrealistic. Every analogy breaks down at a point. "The gasoline engine is like a clock"--this is true in some ways; it depends upon the purposes for which you make the analogy.

The ancient mythologies were realistic to the point to which they were used. When the analogy was stretched beyond its domain, then the society was in trouble. That's why I say that it is dangerous to use these metaphors as a road map without understanding to what extent they are valid.

ANDREE: Whether the myths were realistic is not as important as whether they were believed.

WEIZENBAUM: That's right.

SHADER: And that is one of the difficulties with the aubomobile. At the time the auto came along it was replacing a mode of transportation that already existed. In some sense we are still building horseless carriages. The automobile, then, represented a difference in degree rather than a difference in kind. Thus I think it would be unrealistic to have expected the automotive engineers to worry about the social effects of their invention.

The computer is different. There is not that kind of background to draw on. As nearly as I can tell, it is a completely new force. For that reason I think it is quite appropriate for us to discuss the social consequences of the computer, unlike the automotive engineers.

GRUENBERGER: I get worried, too, when an analogy is stretched to the breaking point. This constant appeal to justify things in computing by drawing analogies to the automobile bothers me. One of the breakdowns in the analogy is that even today a person can sit out the automotive revolution and ignore it. I submit that we are rapidly approaching the point where it is not possible in the same sense for a person to sit out the computer revolution.

FINERMAN: But he can only sit it out in the same way that a person can sit out the automobile; for example, by avoiding interacting with the trucks that bring produce to him or buses that bring people to his town.

GRUENBERGER: Everyone seems to be missing my point. A person cannot refuse to give certain information to government agencies, as much as he may dislike them. To that extent he is perforce going to interact with computers. He can refuse to have anything to do with gasoline-powered vehicles; my only point was that I was agreeing with Joe that analogies break down sooner or later, and the one with the automobile, it seems to me, is getting strained to the limit.

FINERMAN: I believe that previously we reached agreement that some exposure to computers should be given to everyone. I can agree with that conclusion, but I hate to see it confused with the form this exposure should take.

Some people here have suggested that the exposure should be in the form of a general programming course or a language course. Others, myself among them, feel that the exposure should come in the form of attitudes. The emphasis should be on what the machine is capable of doing and what the user must do. There is the notion, for example, of approaching solutions in a logical manner. This definitely places restrictions on freedom of mobility. In return you have the ability to manipulate information,

provided you discipline yourself to live in an environment in which you must be logical. This concept is of primary importance.

ANDREE: Since this problem is so large, each concerned person could help by implementing and carrying out his conviction in his own way. The guy I'm worried about is the one who doesn't have a conviction. Some like to teach appreciation of the computer by using it, which is the way I like and have done. Others prefer to do it without the computer, and I have done that also. Both are fine. If you can get the job done in your way, do it and don't let someone talk you out of it. At one time I believed that computing instruction should begin with machine language. But Charlie Davidson told me that I was old-fashioned, so I tried it the other way and discovered that he was right.

EVANS: I think that Andree is wrong, and I thought so even before he made that speech just now. I'm involved in the education of the hard-core computer scientists, and I think we are moderately successful at that. I don't spend much time worrying about them because I don't think they can fail to become educated in some sense.

Some things disturb me, though. Recently I have been watching the negotiations between a local school district and vendors of packages purporting to furnish computer-aided instruction. Some of the manufacturers are telling the schools that they know all about it and that they can come in, set up the machinery and curriculum, and show them how to do it. The people on the school board are so ignorant of what a computer can do and can't do that they are unable to defend themselves against these sales pitches. If it were not for a few brave citizens who know a little, the school systems would have sold themselves down the river. I think the evidence is clear that the education of programmers and systems designers is not the problem.

ANDREE: You don't seem to be disagreeing with me but, rather, agreeing with me perfectly. I am saying that the problem is so big and the depths of ignorance so abysmal that any technique you care to use is bound to be helpful.

FINERMAN: So you are saying that the problem is not one of teaching programming, but rather teaching what tasks the computer can do and what tasks the human must do.

ANDREE: I'm simply saying that if Charlie Davidson wants to use programming to convey these ideas, that's fine with me.

CANNON: However, teaching programming is a very good way to transmit the right attitudes.

WEIZENBAUM: Whatever the function of hospitals, they should not spread disease, and yet, before Dr. Semmelweiss, that's just what they did. Up to that time (around the turn of the century) they were very smart and very conscientious, but they were spreading disease, and it took Semmelweiss to show them that. Will Rogers pointed out, "It ain't what we don't know that hurts us, it's all the things we know that ain't so." That's essentially the same point.

I think our responsibility as computer professionals, *vis-à-vis* the general public and *vis-à-vis* the social situation, is to come to an understanding among ourselves as to what constitutes truth or reality. In particular, we shouldn't get stuck on what happens to constitute truth or reality at this time. It may only appear to be reality and truth at this time, and its existence may be temporary.

For example, I am sure you would get a consensus among more than 90 percent of all computer professionals to the sentiment that has been expressed here several times that to communicate effectively with the computer it is necessary to be logical and precise. I think that was true and real until recently. A better statement now would be that to enlist a computer's aid economically, it pays to be as

precise and unambiguous as possible. But surely the time is coming when people will be able to converse with computers, make mistakes, say things that are somewhat ambiguous, and eventually in conversation with the machine sort out the ambiguities, as we do in communicating with one another. This notion of precision and lack of ambiguity is being peddled to the public now as a sort of fundamental law. We are continually pointing out that it is necessary to be completely precise and completely logical in dealing with the computer. We constantly reiterate that the computer does only what it is told to do. I maintain that this is a deep idea. For one to understand it usefully, one must understand it deeply.

The slogan, "garbage in, garbage out" which is very much believed for example by *Time* magazine, again needs to be understood in a very deep way if it is to be at all useful. Most people will not understand it that deeply. Our responsibility is to avoid simple-minded sloganeering.

ARMERDING: But in a computer appreciation course, what can you do other than state platitudes and slogans?

WEIZENBAUM: You must do better than that.

KREHBIEL: I think you are sneaking up on taking away all the security that we have by gradually undermining those platitudes that we use and believe. If what you see is true, I can see a gradual withering away of those steps leading to a solution in the problem-solving cycle. It seems to me that all you have left then is the notion of defining the problem as the essential step and letting the computer take it from there. If we get that far, I'm not sure that we even have to define the problem properly.

WEIZENBAUM: Let me bring up for the moment a slightly higher principle. When it comes to writing systems programs, a useful rule that I follow is: If something is useful to the community (whether in a utility program or a compiler or whatever), then the decision as to whether it should be

included should not be contingent on how difficult it might be for the systems programmer to do it. Similarly, if the correct course is to avoid sloganeering to the general public, and we find that it is very difficult for us, that is no reason not to do it.

SANDERS: I think I have a better analogy than the motor car. We live in a computerized society but also in a lawful society. The law is a much better social analogy than the automobile. The law has great influence on our lives, and we are surrounded by it. It affects practically everything we do, and we have been suggesting that the computer will also.

WHITE: The output isn't predictable from the input.

SANDERS: We don't have to understand (most of us, in fact, don't understand--I don't understand) how the law works. I've never studied the law, attended lectures on it, or read books on it. The only time I have to know something about the law is when I come up against it. I suggest that the only time that I have to know something about the computer is when I come up against it. When the card comes through the mail, it has instructions on it about what I should do. Over the years, I--as a consumer of computing or more properly as a consumee of computing--get to understand the kinds of things it does.

We have a mystical faith in people like lawyers and doctors. Their product is valid, reliable, and also expensive (there seems to be quite a close analogy to programming). If we can establish a lawful society, does it not follow that we can do the same thing for computing?

MILLS: The first thing you learn when you encounter the law is that ignorance of the law is no excuse. If you're going to get useful results, then I think the logic has to be there, and you have to go step by step to get the job done. You may make many mistakes along the way, but I

think the general public should know that the computer does just what it is told. The public also should know what the computer is doing to us and for us. I don't think that sloganeering is going to hurt that process. If we could convey the right slogans to the press, it might help the situation a great deal.

DAVIDSON: The need for what I was going to say dwindled rather rapidly as Roger was talking. What I say now amounts to a strong reinforcement of what Roger has said. I think that some of these slogans are highly useful and that it is not necessary to teach all the depths of erudition necessary to understand them deeply. If you talk in schools on the subject of energy and its uses, you don't have to go into mc^2 on the first day. You can help people understand how the world works at a fairly simple level and go into the fine points later. This doesn't mean that you have to teach false notions--it can be put on a simpler basis. With all due respect to the ease with which we will eventually be able to communicate our needs to a computer, they are now at the level of doing what they are told. The computing program that predicts the winner of an election is not a mystical crystal ball. The way the program works is the way that political scientists, economists, pollsters, and programmers have formulated their hypotheses and told it to work. This is one area in which the general public is abysmally ignorant and needs to be informed.

KREHBIEL: We are still largely talking about the large group--the general public--are we not? We still are not concerned with the users or the computer professionals. I go along with the thought that the general public needs to be educated in a precise and accurate way. I think that misunderstanding on the part of the general public can be extremely dangerous. I don't think that the general public needs to be taught computing or problem-solving in the way that we understand it or try to understand it. I think that

they need to be propagandized and indoctrinated in such a way that their attitudes are correct when they come to vote on such a question as a bill on smog.

In a free society this is a spooky concept. I don't know how to handle it, and that's where this discussion has left me hanging. I don't know how you indoctrinate the public with information that is designed to create the right end result but is not necessarily content valid.

GRUENBERGER: Slogans might help.

KREHBIEL: Then I have to support the position that what we want is the right attitude. Perhaps that takes care of the general public. Maybe they should not be indoctrinated in how the computer works and what a program looks like. From the standpoint of the proper attitudes to be adopted by the Salt Lake City School Board, possibly it would be well if they understood that a program never works the first time. Perhaps they should be asking how many iterations the program has been through. I don't think we can resolve questions like these because they have such serious sociological and economic overtones. I would like to divert attention to the users and the professionals in the field. For example, who should be educated at the user level?

FINERMAN: With all due respect to Joe, I don't necessarily believe that because he said these things are taught by sloganeering--by being logical and by dealing in universal truths--that this is the only manner in which one conveys some of these points. I think the same results can be achieved without sloganeering and in the context that all rules are meant to be broken. By and large, these universal truths are true. However, that isn't to say that there can't be exceptions.

WEIZENBAUM: The situation is this. Given a slogan or an assertion in any field, it isn't reasonable to state

that the assertion is true or that it is false. One may correctly say that it is true at this level of explanation and is no longer true at another level of explanation. For example, one can explain how an internal combustion engine works in terms of pistons and spark plugs and exploding gasoline, etc. That is true at one level. At another level of explanation you have to go into molecular theory. The explanations that a computer does only that which it is told to do may be true at one level of explanation. At another level of explanation it may again be true. Most people will learn only slogans, so we must help them to understand that these slogans, under certain circumstances, need to be qualified, even if we can't get them to understand the qualifications because of economics or lack of time.

GRUENBERGER: I wish you would give me an example, Joe, of when the slogan "garbage in, garbage out" is not true.

SANDERS: I'll give you one. A random number generator is garbage in and a simulation might be good output.

GRUENBERGER: Well, I don't understand that one. I can't imagine anything more precise than an algorithmic generation of random numbers. We are bandying words, since I don't consider that as garbage. In fact, I regard a random number generator as a producer of very fine fruit salad.

GALLER: Perhaps a better example would be the analysis of a very large volume of data from which you can draw statistical conclusions that are valid even though the data, item by item, is very weak.

WHITE: A much more valid way of putting it is that output is always dependent on input.

GRUENBERGER: Well I'll buy that, but I don't see how that vitiates the slogan in any way. The intent of the

slogan is better if your input is no good, your output can be no better. I still don't see where the slogan breaks down--at what level.

WEIZENBAUM: Let me put it this way. To understand the computer as a transformer of information (that is, as an input/output device) is a much better way of looking at it and is not much harder to convey.

SHADER: But I would ask where in life and in what area do we not deal in slogans? We always deal in generalizations. There is no such thing as perfect education. It seems to me that if we constrain ourselves in this way, we're making the job unnecessarily complicated and difficult.

EVANS: When you come right down to it, all statements have the character that Joe attributes to slogans. It's not a question of whether a statement is true in terms of its environment, background, and assumptions that went into it. It's a question of the effect that the statement has upon the listener. For example, if you make the statement "this function is not computable," what effect does that have upon the person to whom you make it? It depends largely on his background. It obviously means different things to different people, and yet we are actually getting solutions to uncomputable problems. I think what Joe means is that a given slogan may have meaning to one group and no meaning at all or a misleading meaning to another group, which includes the general public.

GALLER: The word "slogan" is a loaded word, and I think we should move away from it. I think about the situation in which I find myself when I sit on a faculty committee in geology, for example. How much can I be expected to know about geology? I apply those generalizations that I got out of my early contact with geology and try to be aware that there must always be more behind it than a simple generalization. I can't keep all of geology in my head--in

fact, I can't keep up with it at all--but in any field other than my own I have somehow found my own personal generalizations. I think that a less educated person would rely more heavily on generalizations. I would tend to question generalizations more often when I use them. I think we can only hope to convey some of the better generalizations, and if they have to be tricky sounding to stick with people, that's advertising and we'll have to make use of it.

CANNON: I was interested in the remarks suggesting that computer programs which are tolerant of input ambiguities may relieve humans of some of the exactness that most computer systems now demand. Certainly we want to remove burdens from the person where possible. Since we cannot tolerate input ambiguity if it means that the outcome of the computation is in doubt, I assume that what was being referred to included a conversational mode capability in which the true intent of the person at the terminal is ascertained unambiguously by a series of questions posed by the computer program in response to the user inputs. If an input appears not to make sense in terms of the programmed criteria, I suppose the user is asked whether he really means what he says. Thus the computer helps him search to find what he was trying to express. I am all for this type of programming, but only because people are not by nature particularly logical or consistent. If we were, we could dispense with the overhead of a question-and-answer session.

While we will no doubt always expect input ambiguities from humans, the more structured the users can learn to behave, the more efficient and reliable can be our systems. It is still desirable that people be trained in logic, procedure, and consistency. Slogans (or whatever we call them) are generally not sufficient to generate rational habits. This comes only through drill, and the conversational-mode search for the logical content of the input is an excellent form of drill. The computer is not only eliciting the

intended input needed to carry out its primary task, it is also training the individual. Thus it performs a dual role.

There is, of course, a danger in this. If it is true, and I believe it is, that the computer in conversational mode can be programmed to subtly guide the user's thinking, then as educators we must encourage effective use of this teaching tool, and as humanists we must ensure that it is not misused.

I don't suppose a conversational-mode computer program has much more chance of accomplishing undesirable brain-washing than many other forms of conditioning to which the public is continually subjected. Nevertheless, it is still true that the few people who are designing the computerized systems and writing the programs which impinge upon many lives have a measure of control which not only should be under the professional surveillance of specialists such as ourselves, but also under the scrutiny of a citizenry adequately educated to this task. We must not only have the computer to assist people in being rational and logical, but we must make sure that they understand what is being accomplished.

[Agenda Topic 8: What should be done now about training those who will teach computing?]

KREHBIEL: I would like to propose a constraint. If we treat topic 8 as we just treated topic 1, we're going to miss some of the things that need doing. We should keep clearly in mind whether we are talking about training elementary school teachers, secondary school teachers, those in college who teach professional programmers, or those who will teach computer appreciation courses. We need some definitions before we can discuss intelligently.

GRUENBERGER: Let me see if I can define the level. I think the biggest mass market now is the lower level

introductory course for undergraduates at our colleges and universities. At least, that is where we are putting all the money right now.

FINERMAN: Is there any problem?

MILLS: There sure is!

SANDERS: What is the problem?

MILLS: Having people teach programming who don't know how to program.

FINERMAN: It seems to me that most colleges have a natural screening process. The assignment of courses to a teacher who doesn't know the subject well is usually based on emergencies of the moment. However, I believe that the situation in elementary and secondary schools is quite different since the people who spread false information simply don't know any better. The reasons for misassignments in the colleges and in the secondary schools are vastly different. The secondary schools do not have the same excuse for lacking information, such as not having enough money to have access to a computer. They simply don't know any better in many cases. Not only are the teachers in the elementary and secondary schools ill-prepared, but they may have no interest in the subject matter. We can say that a college teacher should know the subject he is teaching, but I don't think our saying it will have that much influence. If we could come up with something meaningful for the high schools, I think we would have a larger influence.

MILLS: I have to agree with him that we can obtain the greatest leverage by producing a workable plan for the high school and elementary school teachers. In too many cases, the colleges have been sold a bill of goods by someone who claims to be a teacher of programming.

I am keenly aware of this because I happen to teach at night in an adult program, and the only requirement was to

apply. I'm afraid that we have many people in the Los Angeles area now teaching programming who couldn't program their way out of a paper bag. I agree with Aaron that the payoff comes if we can get the message through to the teachers in the high schools, the junior high schools, and the elementary schools.

FINERMAN: I'd add one more category to those that Roger listed, because it is different from all the rest--the training and education that takes place in our junior colleges, the two-year schools. They provide an in-between world, between the high schools and the colleges.

ANDREE: Some junior colleges are excellent, college-parallel, two-year institutions; some are excellent trade schools; but others are little more than havens for almost-dropouts. There are all kinds.

KREHBIEL: I'll have to speak on behalf of the junior colleges. We generally deal with a broader spectrum of talent than does the four-year college. This means that we must do several jobs to meet the varied needs and talents of the students. The problem is unique and should not cause a detour here. I agree that the high school is a very productive area and the one we should consider.

SHADER: If you are going to talk about reaching the high school teacher, you have both an advantage and a disadvantage. You have a greater opportunity, because most of our high school teachers are trained in the teacher colleges. They are bulked together in one place. You'll have to put up with an enormous time lag, which I suspect is a full generation, to get the job done. I think it would be almost impossible to retrofit the teachers who have had a number of years out in the field. There are exceptions, of course...

GRUENBERGER: We'll show you some of them after lunch.

SHADER: ...By and large, the focus must be on the schools of education. But they have been the slowest to move into computing of all the institutions that you can categorize. The junior colleges, the four-year colleges, and the universities have been moving into computing at a rate that might actually surprise you. But for the teacher colleges it has been virtually nil. We should hear from Atchison on this point because he is somewhat of an expert on the subject.

ATCHISON: I agree that little is taking place, but I think that there is some activity and it is encouraging for the future. There is, for example, a course at the University of Minnesota designed by Dave Johnson for the training of teachers. He has written it with my encouragement and submitted it to the *Communications*. I agree with Mel that this is one of the areas urgently needing encouragement and advancement and that is why I persuaded Dave Johnson to write a report about what he has been doing.

At the same time, I think that those colleges and universities that are offering formal courses in computing science have an obligation to tailor some of the courses to the needs of their adjacent schools of education.

A third avenue of attack is through summer institutes for teachers. I don't know how successful Andree's institutes have been, but I know that some institutes have been notably unsuccessful. I think we have an obligation to help some of these summer institutes become more successful.

ANDREE: We have been lucky. The people who come to Oklahoma in the summertime come because they want to.

DAVIDSON: There's a point of order here. Are we discussing the training of high school teachers that will enable them to give courses in computing, or informational courses that will help them interject computing ideas into the other subjects they teach?

MILLS: Yes, both.

GRUENBERGER: Let's back off a minute. I'm in agreement with Krehbiel that we have to focus this discussion, or we go off in all directions. Let's try to limit it to what we can do or suggest for the training of high school teachers while they are training to become high school teachers (of any subject).

FINERMAN: So we will try to keep it independent of the discussion of whether computing should be taught as an independent subject.

ANDREE: I think it is perfectly proper for us to discuss how a civics teacher or a political science teacher should bring computing topics into his course. The fact that computing and computers are affecting the civics of today is something that this teacher should know. As far as I know, no one is trying to give either future civics teachers or present civics teachers any of this knowledge. If someone is doing this, I would be very pleased to know about it because I have some civics teachers that I'd like to send to that course.

FINERMAN: But this approach is based on what might be a misconception; we are assuming that we can get to these teachers while they are attending teacher colleges. I don't know how the educational system works in California, but in New York State the trend is definitely away from teacher colleges. The person who is training to teach physics in high school in New York State enrolls in a physics program in college rather than an education program. These future teachers are graduated as physics majors rather than as education majors. They may, of course, have taken some courses in education. However, my point is that you can't capture these people at a focal point called teacher colleges as you once could. You can't identify the future teacher as easily as you could ten or fifteen years ago.

MILLS: California has come around to doing it the same way as New York.

I want to comment on a remark that someone made--that you have to catch the teachers while they are in training, or it's too late. We finally broke through to the Los Angeles School System (as you'll see) and conducted a workshop for some of the teachers. This can go on all the time. It's not too late--even for those who have been teaching for a long time. I think you'll find the right kind of teachers are looking for this sort of thing, and in the case of the workshop that we ran, more than half of the teachers didn't even need the points. I think it's true that there is a great demand for such training on the part of many teachers. The civics teacher, for example, is bound to notice that computers are involved in election returns, and he would be naturally curious to know more about it. There may be some question about the depth to which you can go in such workshops, but I don't think there is any question about the need for them.

ATCHISON: We should recognize that there is a long-term problem involved here and an immediate problem. I was addressing myself before to the long-term problem, and the teachers involved are now in school. Our teachers do get their training in college and, therefore, I think it is necessary that something be done to reach them now so that they are equipped to teach the things we have been talking about.

As far as the immediate problem is concerned, the teachers that are now out in the field are being required to teach various aspects of computing in math courses and others. Therefore, we should concentrate on holding workshops for them.

WHITE: It is generally true, as Aaron pointed out for New York (and I can verify that it is also true in California), that those who will be teachers do not major in

education. It may be difficult to identify them as teachers, so it's essential that we catch them in workshops after they have identified themselves. Hence, the in-service institute may be the best place to transmit this information to the teachers.

CANNON: I'm not sure it's the best place, but it's one of the few places now. It's been my experience that in our summer courses in computing, we have many teachers coming in, along with engineers, people working on their Master's degrees, and so on. The more experienced they are, the more difficulty they seem to have with basic concepts.

One summer, in one of these courses, two of the students brought their little brothers along who were junior high school students. I remember that when I explained some particularly difficult concept, there were dazed looks all around the room, except for the two kids. They would impatiently nod for us to get on with the material. Incidentally, at the end of the course, they placed above the ninetieth percentile. They did better than most of the teachers and came out ahead of a very successful engineer in the aerospace industry who was working on his Master's degree. It seems pretty clear that if you can catch them when they are young and flexible, they can take these things in stride without going too deeply into the fine points. When you're dealing with people who are older, particularly people who are successful because they can interrelate things, it's difficult to keep them at the level of 1 plus 1 equals 10 because they are off in the wild blue relating things that you don't want them to relate at the moment. This goes back to our discussion of Agenda Item 1, with the conclusion that we should begin to instruct the young while they are still pliable. Of course, the more experienced people show much more mature insight into how these tools may be applied.

SANDERS: The trouble with that is that the people who teach them when they are pliable are unpliable. The obvious answer is to have kids teach kids.

ANDREE: We've been exploiting that for quite a few years now in our in-service institutes for high school teachers. We pick the high school teacher, and she picks one, two, or three of her students to come along with her, and we teach them all computing and related mathematical topics. We have found that if we try to teach the teachers all alone, they fight back with the attitude that the "ivory tower" professors just don't understand them. When their own students are there with them, then they simply have to understand it--from humiliation, if nothing else. It may be dirty pool, but it is effective.

EVANS: We've been playing the same kind of game.

GRUENBERGER: Yes, this story has been repeated at many places around the country. It's part of the philosophy that many of us have observed. When you are dealing with children and the computer, you couple the child and the computer as soon as possible; then your main job is to stand back. Not only is there a teaching link between the machine and the student, but, of course, in groups the children teach each other in some mysterious fashion we don't understand.

EVANS: We have, however, dealt with high school kids alone in summer institutes, and this has had a pleasant reaction because the teachers now say, "I need help."

MILLS: I think that's dirtier pool than what Andree is pulling.

GALLER: I wonder if this is a thought that should feed back through you, Bill, in your work in NSF institutes; namely, that each teacher should be required to bring a student along.

ANDREE: I hope it didn't sound as though I were criticizing the teachers; they are very important to the venture. They really add a great deal. The teachers not only have maturity, experience, and specialized knowledge, but they know where to go for information. Those with whom we have dealt largely have Master's degrees. But it is the case of the team being larger than either of its parts.

GALLER: In addition, the teachers can maintain continuity and keep the activity going at the high school.

SHADER: That's right. It's only through the teachers that you can get the multiplying factor you want.

CANNON: We should also keep in mind the panic (which I have noticed even among college professors) over being rendered obsolete. We must try to allay that fear and assist them in getting over their panic. Mainly, I would guess they would need assurance that this subject can be learned.

ANDREE: One thing we have done is to have an eighth grade student serve as a lab assistant for the high school teacher institute. He is not only an interesting kid, but having him around is beneficial to the teachers because this gives them the impression that if a 13-year-old boy can learn it in one year, then, so can they.

SHADER: What we are saying again is that the problem does not lie with the kids. They are enthusiastic and highly motivated. We don't have to worry much about them. The problem goes back to the teachers.

GALLER: But you can't bypass them.

ARMERDING: So we are dealing with a short-range problem because with another generation, the kids will grow up and be the teachers, and the problem won't be there.

SHADER: I think you're right--it's a generation problem, provided we do something about it now. Otherwise, it will be a longer-term problem.

KREHBIEL: If the teachers would give the students the right kind of problems and require the right sort of solutions, then as we pointed out before, the main thing we can do is get out of the way. It seems to me, then, that what is needed is a collection of the right kind of problems and an insistence on the right kind of solutions.

GALLER: But you have to get the teachers willing to do it.

KREHBIEL: Yes, and it is fear that is keeping them from doing it. It's also partly lack of knowledge. I have had physics teachers come to me and ask what kind of problems they can assign. We've gone at this one through the students by telling them that whenever they see a problem in one of their courses that can be solved on a computer, they should solve it that way and give their instructor the solution. This had a rather violent reaction too. Gradually, we are noticing its effects around the campus. The teachers are beginning to assign problems which should be done on the computer.

SANDERS: You're saying, then, that you're using the students to teach the teachers.

WEIZENBAUM: It is probably essential that the teachers be trained to defuse and debunk the computer to some extent. We must reduce the anxiety that the teachers have about the computer because of the attitude that it is too complicated for them to learn. The only way to do all of that, it seems to me, is to show the teacher that he can learn it. Frequently, what gets in the way is the social situation that makes the teacher embarrassed. One possible way out of this, when the resources are available, is a teaching-machine program. The computer itself is still the best teacher of

computing. The machine is there, and it is better suited for the teaching of computing than it is for chemistry, civics, or whatever. We have such a computer program at Tech for the purpose of teaching programming (and I emphasize that it is *not* to teach FORTRAN). People can link to this program somewhat anonymously at any hour of the day or night off in a corner. It doesn't have to be done in a class. We are able to record who uses the system, and even though it is only a few months old, we find that many members of the senior faculty have tried it. It seems to me that this goes a long way toward defusing and debunking the computer. They are acquiring the kinesthetic experience of actually communicating with a computer and doing it successfully.

ATCHISON: I want to reinforce something that Krehbiel brought up, which is this query on the part of teachers on what sort of problems are suitable for the computer. I have met this query, too, from teachers and as a result, there will be a little department starting in the *Mathematics Teacher* in February or so, that the mathematics teacher can use for this purpose.

MILLS: Armerding indicated that the problem will go away in a generation because the kids will then be the teachers. But that still leaves one small problem. We're not reaching many of the kids. Somewhere we must get the ball started.

FINERMAN: We'll have to make sure that the kids now going to college are exposed to the necessary material.

ANDREE: But the computers twenty years from now will be so different that I can't begin to imagine how they will look.

EVANS: It would be interesting to know how many college students are being exposed to computing. At our school it is roughly one-fourth of the freshmen.

FINERMAN: Hamblen has some figures in his survey which, if I recall them correctly, indicate that approximately ten or eleven percent of all students now going through college are getting some kind of formal computer training. The Pierce report gives a figure of a little over five percent as of 1965, which is not inconsistent.

WHITE: The figure sounds as if it is less than all of the technical people in our colleges. It's a cinch that we're not hitting the people in the social sciences and, in particular, those who might be teachers in the social sciences.

WEIZENBAUM: But we are in an exponential growth situation here, and our extrapolation in this particular instance has to be Grosch-like; namely, over-optimistic. The statistics I could cite for MIT are not typical, but I do think they are a herald of the future. Nevertheless, I have noticed that a significant fraction (I don't know the figures, but I am sure it is significant) of the kids entering MIT now have a good knowledge of computers before they even get there.

SHADER: What are the statistics at Wisconsin where they have probably one of the oldest programs in the teaching of computing anywhere in the country?

DAVIDSON: There are always from 700 to 800 students taking one of the four first courses in programming; these offer four different approaches. This is not a very high percentage out of 33,000; the program went through a period of doldrums for some time. This last year, however, the department grew 30 percent, having the greatest growth of any department in the university. So even though the exponential growth Joe mentioned is entering in, it is still a tiny fraction of the people who might be exposed.

SHADER: We seem to be intermixing two subjects here; namely, the teaching of computing *per se* and the use of computing techniques to teach something else. I'm interested in the latter--not how many are learning the basic skills of computing, but to what extent it is infiltrating other courses on the campus.

DAVIDSON: All the engineering students, numbering about 3000, are exposed to computing. Students in the sciences are not. The 700 or 800 that I mentioned includes our computer appreciation courses as well as three other courses, including one in numerical methods.

FINERMAN: Our experience in New York confirms this. Our engineering students all take a basic introductory (not appreciation) course. Students in the sciences, including physics and chemistry, may or may not take this course. Oddly enough, despite the lack of interest in computing shown by the mathematics department, the students in mathematics, particularly those who intend to teach, generally see that they get a computing course. However, outside of certain engineering courses, computing is not infiltrating courses given in the particular disciplines. Most infiltration is taking place in the social sciences rather than natural sciences courses.

EVANS: At Utah we counted twenty-four departments in which the students were assigned homework that required computer solution.

DAVIDSON: An interesting aspect is that next year the School of Business at Wisconsin will start requiring a computer appreciation course including programming as an entrance requirement. That should add another 600 students a year.

ANDREE: At Oklahoma we have a course in what we call "baby statistics." No one in business, science, or mathematics takes it. It has no calculus prerequisite. It's

not designed for the rigor that the math and science people would normally take, but it does require hands-on use of the computer throughout the course. It seems to be going very well. Psychology, social science, and biology students seem to be taking it.

CANNON: As it said in the President's report, our growth has been so phenomenal that we have almost reached our infancy.

GRUENBERGER: Ten percent isn't bad, particularly since it has grown in the last couple of years.

SHADER: It has enough critical mass, so I think it will keep going. I would like to come back to the high school level because there I don't think we have a critical mass.

WHITE: The critical problem is finding problems. When we teach a course, we can usually find plenty of problems in our own area. The difficulty is to find suitable problems in other disciplines.

ANDREE: Fred Gruenberger has a nice little book, published by Wiley, that has 92 problems in it.

SANDERS: Maybe we should discuss another one of these Agenda topics; namely, "What can be done to get better textbooks?" Maybe we don't need so many books on programming, but more on spheres of application.

GALLER: Maybe there is a role for ACM here to commission people to collect different problems in different areas. These should be problems that are specifically designed for teaching purposes. In our project in engineering at Michigan, the entire output was several volumes of problems in various areas. The books contained solutions also.

GRUENBERGER: Where was this distributed?

ATCHISON: That is one of the real problems. The output Bernie mentioned was fine, but the distribution was poor.

GALLER: It was at the college level, and the final report was sent to every engineering professor in the United States.

MANY VOICES: That's fine, but I didn't get it.

GRUENBERGER: Was there a notice in the *Communications* that this was available? How would anyone, outside of an engineering professor, even know it exists?

MILLS: Besides all that, may I point out that more teaching in programming has been done outside the colleges than inside. By and large, industry has done most of the teaching. That's where most of you learned to program. I would guess that 90 percent of the teaching has been done outside the academic atmosphere. In fact, I would guess that Lockheed, North American, Douglas, and Boeing would account for three-quarters of the programmers in the country. That was certainly true as of 1956, and that is the generation all of us represent. Are the experts that we have now the people who have been in the business for the last year or two? I don't think so.

MANY VOICES: Yes, those are today's experts.

KREHBIEL: We seem to be all mixed up, and I'm lost again. Much of the discussion in the last few minutes has been on the teaching of programming, and we are dealing again with the professionals. As I see it, the subject of programming is a subset of the subject of computing. I think we have been talking largely of programming courses that may or may not begin with machine language, but usually wind up with FORTRAN. I want to talk about courses in computing in which words like "heuristic" and "stochastic" are mentioned. I want to talk about courses in which problem

solving is the central motive. To me, a course in computing allows the student to see a broad spectrum of applications and uses. I want the student to see how problems are solved and to see the techniques available to solve them in simple form. I would like to see a relatively small portion of such a course devoted to the programming of any given machine in any given language.

I don't think that is what we have been talking about. From what was said, it sounded to me that what we are talking about was coding courses.

MILLS: Most of the courses in our colleges are coding courses.

KREHBIEL: I am familiar with Andree's book, and I know he isn't playing that game. When you talk about Douglas, North American, and Lockheed, you are talking about how to learn to code.

MILLS: That's right. And you would be surprised how many coders became programmers. You might also be surprised how many didn't. The latter is especially true nowadays. I'm agreeing with you, and I'm happy to see that the colleges are now beginning to do a beautiful job in teaching computing--that's what they ought to be doing.

With the distribution of these volumes that Bernie mentioned, it rang my bell--we never heard about it. I happen to teach programming or coding at TRW Systems, and I'd love to have those volumes. I happen to have Gruenberger's problems book, but that's just because I happen to know Fred.

GALLER: I'm making a note to see if those volumes can't be resurrected somehow and made available to you.

GRUENBERGER: Bernie, just exactly what are we referring to here?

GALLER: The reports we are talking about are the first, second, third, and final reports of the Ford Foundation Project on the "Use of Computers in Undergraduate Engineering Education, 1960-1963" at the University of Michigan.

[Editor's note: The report is available on microfilm from University Microfilms.]

FINERMAN: I've made use of these volumes, and it is my opinion that there is one principal difficulty; namely, that while the problems may be very good, some of the solutions are very poor indeed. In many cases I feel that in presenting a problem and showing how to solve it, the main point has been completely missed. In many cases I've found that the solution is one that you would use with a desk calculator. There is a complete lack of understanding that an algorithm, developed for computer solution, may differ fundamentally from the algorithm intended for desk calculator solution. The model, the techniques, the whole approach may be quite different.

GRUENBERGER: I don't understand how you can have the only solution to a computer problem at all. The solution at best would be a flowchart, but more often than not, there are many solutions, and in presenting even one you have frozen the class's thinking. It has never ceased to amaze me that when I give a problem to a class for homework or a quiz (for which I think I have a neat, efficient solution), I find that I then have 28 solutions, none of which were the one I had in mind and most of which are significantly better.

GALLER: Of course, in justification it should be pointed out that much of that work was done by people who were learning at that time.

But I don't want to push that point. This all started from my original question. Should there not be a project

by ACM or some other agency to commission people to look for problems and systematically promulgate them, not only in computing but in the other disciplines?

GRUENBERGER: That is exactly how my problems book got started. We found at a meeting in Washington that quite a few people had collected a few problems. By systematically putting them together we felt we would all be richer. I have been trying to collect problems like that for years, and I am anxious to obtain any problems that I can. That's why it appalls me that there has been some kind of systematic collection around for over four years, and I have never even heard of it.

FINERMAN: I agree there should be such a project of gathering problems, but it should be monitored and controlled. Someone well-trained in computing should look at the problem in sufficient depth to make sure that solutions presented are not spreading misinformation. Otherwise, I fear we may do more harm than good. This is akin to the notion, spread by many, that understanding a computer language means understanding the use of computers for problem solution.

GALLER: What you are saying to me is that if it should be done, it should be done right, preferably by a team of at least two people, one representing the particular discipline and the other representing computing.

GRUENBERGER: I'd like to reinforce what Finerman just said. I happened to stumble across some work being done in our computing center by a student who was enrolled in a statistics course. I found to my complete amazement that he was busy calculating standard deviations by running through his data to find the mean, differencing every item from the mean, squaring, and so on. In other words, he was doing it all wrong. And he was doing it that way because

his teacher had not come into the computer age yet but was still operating in the desk calculator mode *à la* 1935 or so. I pointed out to the student that we have better formulas for calculating the standard deviation that require going over the data just once (namely, the computer way), but he was averse to using this solution because he had been told in his statistics course how to do it. That's what I mean about giving solutions. If you are going to give them, at least we should give the best one we can find, and one that is geared to the computer. But I'd rather see no solutions at all. We faced this problem in our problems book by presenting at best some flowcharts or maybe some hints but constantly stressing the theme that the student can always find a better way of doing it.

DAVIDSON: I'm concerned that we worry so about doing a perfect job that we may not get started doing any job at all.

WHITE: The whole project seems very timely, and we should get started.

ANDREE: While we are on this subject, we should get in the record a mention of the book that just came out, written by Ben Noble, *Applications of Undergraduate Mathematics in Engineering*, published jointly by the Mathematical Association of America and the Macmillan Company in 1966. It merits your attention.

SANDERS: Is there anything in print on situations where you can't use computers? Isn't this equally important? If we don't do this, can't we go overboard and present applications to students to lead them to believe that the computer can do anything? Students are good at extrapolating on the basis of no experience. There are many problems in areas of thought to which you cannot apply computers at all, at least not today.

WHITE: Yes, but it is important to qualify that. We are getting too many examples of situations where we say that something can't be done, and these young kids go ahead and do it.

GRUENBERGER: Let me add another outstanding example of the thing that Aaron was talking about. The first edition of the SMSG book on computing (intended for high school students) introduced the students to problem solving by a homely example. This was intended to be a simple example that they would surely understand; namely, how to calculate the gas mileage of an automobile. They presented it in the form of an algorithm for which step one was "empty the gas tank."

That is exactly the wrong way to go about that problem; it is utterly stupid. If anyone starts to teach problem solving and demonstrates to me so forcefully that he doesn't know how to solve a problem (even a simple one), then he has lost me for the rest of the book.

DAVIDSON: Can't the problem be solved that way?

GRUENBERGER: Well, it either introduces interesting new problems in arson, or you have to learn how to turn the car upside down. In either case I say it is wrong to go about it that way. You can, I suppose, bail out the ship with an eyedropper, and I imagine you can open a can of soup with a darning needle in your quarter-inch drill. But those are not the proper solutions. If we are going to show people how to use the computer to solve problems, we owe it to them to show a few examples of how to solve problems right. Showing students bad solutions (even if they work) is simply fostering bad computing.

FINERMAN: I wouldn't worry too much about a desire to do a perfect job and having that hold us up from doing any job at all. However, I submit that doing a job where perhaps 50 percent of the cases have the wrong solution model

or technique is spreading bad information instead of good. I'm saying that we shouldn't rush in simply to do the job and, thereby, do the job wrong. We must be sure we are not spreading misinformation.

WEIZENBAUM: The danger is that the good will drive out the best. We have another danger--we're being much too conservative. Everything we have dealt with in the last few minutes has been in the area of numerical examples, and we have been making a tacit assumption that those of us who have been in the field for many, many years are the leaders. I don't agree with that. I think the young people coming in will be, if they are not already, the leaders. The point is that we grew up considering the computer as a numerical calculator. I think it is important to teach young people now to regard the computer as a symbol manipulator, with numbers simply being a set of symbols that have special properties. We need many examples in non-numerical areas.

KREHBIEL: And I would argue strongly that we should not present solutions at all. For example, I know nothing about sociology and haven't the faintest idea how a computer could be applied to problems in sociology. I'd like to know. I'd like to see a dissertation on the application of computers to sociology problems. I can think of problems in some areas, but not many. I would venture to guess, though, that if we could find the problems, the kids would find the solutions and good ones.

WEIZENBAUM: It turns out in many cases that the way you arrive at a *good* solution with a computer--as opposed to simply finding a solution that works--is to find the appropriate representation. If you just have a bunch of numerical problems that are solved in floating point regardless of the appropriate representation, then you are missing a large bet. The difficulty that people in the

social sciences have is in understanding that they don't have to adopt as a first step some sort of numerology. They must be made to understand that other representations are possible and that they can invent representations. They should be encouraged to devise representations that might appear funny to the numerologist (I use that term to stand for the numerical analyst).

KREHBIEL: There is a danger also in providing him with any solution. I think we should just let him experiment with the computer.

GRUENBERGER: Do you want him to try to empty the gas tank?

KREHBIEL: Well, yes, as a first cut at it.

WEIZENBAUM: Let me give you a good pedagogical example. Unfortunately it is numerical. I tell the student that I have four integers, A, B, C, and D, which I am going to relate with the equation

$$A^n + B^n = C^n + D^n$$

I will supply the value of n, and I want a set of distinct integers A, B, C, and D that will satisfy the equation. If we are going to do it at all, n will have to be small, but I don't propose doing it; I simply want to talk about how to do it. I can imagine a program for which n = 5 will consume 50 hours or more on a 7094. I have done this (not run it on the machine, but devised a program), have gone to work on the program to see where I could shave it down, and have succeeded in cutting it by four or five orders of magnitude. Finally, I can get it down to where you simply have to push the button, and the four distinct integers will fall out. Here is something you can do without actually using the computer, and there is something pedagogically valuable about it.

ANDREE: I'll give you another example. We have been experimenting with the Euler phi-function (or rather the inverse phi-function). There are formulas for the phi-function, but it turns out that that is not the way to do it with the computer. One of my students came up with a sieving technique that cuts the solution down from what we anticipated would be 3000 hours of 360 Model 40 time to about two hours.

GALLER: I think it is vital that the people who propose to teach the teachers acquire an understanding of data representation. It's part of the problem. When I started a new course recently at Michigan for people who did not have mathematical backgrounds, we spent a considerable amount of time on the subject of data representation and then wound up with a few problems. These were people from music, zoology, law, and library science. I think this is one of the few things that all of them needed to know.

CANNON: We talk about problems to solve, but we should realize that this is not necessarily an end in itself. It is, to be sure, a tremendous tool for teaching. The goal, however, is to convey generalized concepts, such as those involved in data representation. The subject is not very well-structured to do this as yet. To accomplish this, it becomes necessary to devote a substantial effort toward breaking down harmful stereotypes that exist in the minds of the people we are teaching, and perhaps sometimes even in our own minds.

Let me illustrate the stereotype problem with a classroom in which the students play out the roles of various cells and registers in the computer. This comes at an early stage of their instruction. They do understand, or at least have been told, that the computer will carry out exactly what it is instructed to do. We eventually encounter the instruction "make the accumulator negative,"

but I see to it that the accumulator sign is already negative at that point. Three-quarters of the class generally want to set the accumulator positive. I ask them how they can possibly derive out of the three words "make accumulator negative" the operation that will make it positive. It comes as quite a shock that they cannot jump to such pat conclusions.

There are various attitudes and conceptual relationships that must be taught. We should bend our efforts toward these goals rather than simply searching for appropriate problems to be solved. We expect students to derive their own conceptual framework from routine programming chores that we give them. We force students in this area to abstract meaningful relationships from their computer experience. It takes more experience and insight to succeed well in this endeavor than most students can muster. As educators, the duty falls upon us to do the abstracting and conceptualization for them. We must guide them in mastering a logical structure of concepts in the same way that mathematics or other well-established disciplines are taught.

GRUENBERGER: I think it is terrible to collect problems without giving some clue to proper solutions. I think the example I cited awhile ago (of students calculating standard deviations the wrong way) is typical of what I have in mind here. The students were simply doing it all wrong.

DAVIDSON: But were they getting the mean and standard deviations they wanted?

GRUENBERGER: You keep citing this pragmatic approach, Charlie. You can reduce the fraction 26/65 to lowest terms by cancelling the 6's, too, but I can't accept right answers as justification for poor (or wrong) methods.

This is one thing that we can do to reduce the confusion and mystery about computers. They can be programmed stupidly,

just as we can operate stupidly in other areas, but since computers let us play out the consequences of our actions, they are unparalleled in letting us play out the consequences of some intelligent thought, too.

DAVIDSON: But maybe through that they'll discover the reasons for doing it in a better way.

GRUENBERGER: But who is ever going to show them the better way if they are told bluntly to do it the wrong way? They're quite happy to get their answers by chewing up abnormal amounts of machine time, and we don't have that to spare.

DAVIDSON: By having students approach problems in various ways, you are going to discover which are the good solutions and which are the bad solutions from your efficiency point of view. I contend that there are not poor solutions from the standpoint of learning how to use computers.

GRUENBERGER: And I submit that any solution that chews up 20 to 100 times as much computing time as it should is wrong. I don't object to the students doing that once, but someone should be around to point out to them that it is wrong.

[After lunch, the AFIPS film "It's Your Move" was shown to the group]

DAVIDSON: Before lunch, we skirted a question that I'd like to bring up now. We talked this morning mainly about the question of training high school teachers in an understanding of computers so that they can refer to them more intelligently and relate them to the teaching of their own subjects. This was apparently also the goal of the institute shown in the film. I would like also to discuss

the concept of courses in high schools specifically devoted to the computer. Are such courses desirable, and if so, where do they belong? This is a question that I keep getting over and over from high school teachers.

WHITE: We get the question not only from high school teachers but from administrators, and usually in the form "If we are going to put such a course in, what do we have to throw out?" They are saying that their curriculum is packed now, and they are asking what such a course is more important than.

GALLER: First of all, there are many courses in the high school curriculum that are optional or elective. But I don't think that's the way to go.

WHITE: I don't think that's quite true, Bernie. The courses may be labeled elective, but the college-bound student, at least, has to fill up every class hour with courses to some extent that are required for college entrance.

GRUENBERGER: I used to believe that, too, until I started examining the final transcripts of some of the kids who do arrive in college and it turns out that there are relatively few subjects that in my day were considered "hard" subjects and quite a few courses that truly can be considered electives.

SHADER: I think the proper statistics are that 60 percent of the high school population goes on to college and 40 percent does not. For the 40 percent not going to college, it might be proper to have a course in computing. This might be in the nature of vocational training.

WHITE: But probably very few in that 40 percent could get through a course in computing.

SHADER: I'm not so sure. At the conference that Atchison and I attended in Denver a couple of weeks ago, we heard a report from Paul Rosenbloom about experiments he is

conducting in Des Moines, Cleveland, and New York. He is at Columbia and is using U.S. Office of Education money. In New York he is dealing with the so-called 600 schools, which are those involving the kids who are alienated from society, the extreme under-achievers, the discipline cases. These kids take no direction from anybody and are completely uncontrollable.

GRUENBERGER: How do they get the kids to school every morning?

SHADER: By force--literally. They have had some fantastic results in using the computer as a motivating factor for these kids. It would take some time to describe their technique, and I would leave that up to Atchison who is the professional educator, but I would comment only that they have had outstanding success in three widely different cities with different kids. Not only have they had success with computing, but also with computing concepts (such as flowcharting) in the teaching of basic arithmetic. The children involved are in grades 7 through 10 and have never learned arithmetic. They can't do the basic arithmetic operations. If you try to retrain using the same technique they had in the fourth grade, they simply won't pay attention. But by using computing techniques and the computer to convey the concept of arithmetic, it seems to attract their attention. There is another difference between these kids and bright ones. When they do learn some skill, they seem to be quite happy using it over and over. The under-achievers seem to get a great deal of satisfaction out of doing over and over anything they have learned to do. The results that they have had here seem to be incredibly good.

What reminded me of this was your remark about the kids who were not going to college. I have a feeling that you can attract them in a meaningful way with the computer. You would be showing them something that they could learn that

is meaningful and relevant to today's society. It would probably be more difficult to teach these children. As Rosenbloom has pointed out, as a general rule the teachers who wind up teaching the under-achievers are themselves the under-achievers, but the whole program has had enough success to justify going with it.

FINERMAN: You have been talking about using computers to teach.

SHADER: No, that is not so. Let me explain with an example. To teach multiplication, you raise the concept of a little black box that has two inputs and one output, and if you put a 3 on one of the input lines and a 4 on one of the others, then a 12 appears on the opposite line. The kids seem to respond to this approach where they would not to a simple memorization of a multiplication table. Then you begin to put these things together in sequences of operation. Without too much difficulty, you can transmit notions of association, commutation, and much of the logic of basic algebra.

Just as an aside, they have claimed that these classes involving the computer have had the fewest attendance problems of any of the classes given the under-achieving. The kids do not have to be brought to these classes by the truant officer.

WHITE: These classes, Mel, are not taught by under-achieving teachers, are they?

SHADER: No, and they aren't taught by Rosenbloom either. His students and the people working for him are teaching these classes. He formulates the ideas. He happens to be an outstanding mathematician, also, which helps to give him prestige and authority in the mathematics community. The point is that they are having outstanding success, and I was quite amazed to hear about it. I wouldn't have believed it before I heard the facts.

ATCHISON: It is a new teaching technique that seems to be quite successful.

MILLS: Two years ago we had a similar report from a teacher in the Vallejo School District; namely, using a computer as a motivating device for the under-achievers.

SHADER: Perhaps I should have mentioned that they don't actually have a computer. They are only using the concepts of computing.

WEIZENBAUM: Does anyone have an hypothesis as to why this works?

SHADER: I think it is simply a question of motivation.

MILLS: The main reason is that it's impersonal. When a computer tells you you're wrong, it doesn't have the same emotional impact as a human teacher telling you you're wrong.

DAVIDSON: But there is no computer involved here.

SHADER: It is probably because it is relevant in today's society. It is glamorous, and there may be a certain amount of Hawthorne effect--these kids are receiving special attention.

WHITE: That last statement may account for much of it. You have special teachers, and you are giving the children special attention. It is possible that with very competent teachers giving special training to kids, you get similar results without mentioning the computer.

WEIZENBAUM: I have an hypothesis as to why this works. I think it is urgent that we begin to understand why people get hooked on computers. Why is it that people who won't pay attention to anything else for more than ten minutes at a time, if that long, change their attitude when they approach the computer? Some of these people become programmers, and we have all had experience with a few of them who

work very long hours, neglect eating and sleeping, and devote their full attention to getting their computer program running.

Further, when they finally succeed after all these hours and obtain the program that they started to write, there is a disappointment. This would appear on the surface to be a contradiction.

I think it is urgent that we begin to understand that this phenomenon and the one that you were discussing with the under-achieving children are closely related.

FINERMAN: And what is your hypothesis?

WEIZENBAUM: I don't think it is relevant here, and it would take too long to explain it; however, I'll give you a clue. If I had described the phenomena without making reference to computers, then you could just as easily apply the hypothesis to gambling. It seems very clear that for compulsive gamblers (those who are addicted), winning great sums of money is not the object of the game. Gamblers who win and then quit for the night are disappointed. Gamblers are disappointed that the game is over--not whether they have won or lost. My feeling is that if we understood why this addictive business works, we could use it.

DAVIDSON: To get back somewhat to the subject, should there be specific courses in computing in high school? If so, in what department should they be given and with what goal in mind?

GRUENBERGER: It might be helpful to relate some of the history behind the film you just saw.

As someone mentioned, the Los Angeles School Board resisted all advances by ACM and others for about seven years--advances that offered them anything they really wanted toward the advancement of computing in the schools. Approaches to the school board by vendors are easy to understand--they wanted to sell machines to the Los Angeles School System.

But it is not so easy to understand the negative reaction that the professional groups of various kinds received when they made offers to run courses or do almost anything that the school board would ask to advance the cause.

When they came to the ACM education committees last January, they had just two questions that they wanted answered: 1) What computer should they buy? and 2) What language should they use to teach in? There was no question in their minds as to where it should be. It should be in the math department in grades nine through twelve and should parallel, paragraph by paragraph, the present math curriculum. They had no doubt that someone could magically create worksheets for them that would relate the computer to every single topic in the existing four-year math program.

We had many meetings to convince them that though their first two questions might not be completely irrelevant, other problems were more important. Then, we also had to convince them that it is not going to be possible to relate the computer to every topic in the math curriculum. And, even if it were possible, we wouldn't recommend it. We argued long and loud that if you imbed the computer in the high school math department and ignore the other departments, you have lost half the battle before you get started.

Convincing them on this last matter wasn't at all easy. We actually got down to the point of taking the algebra textbook currently in use and challenging them to find some topic in the book for which we could create a meaningful computer lesson. We discovered that in an algebra textbook at the freshman level in high school, there are few topics to which you can apply a computer. The bulk of the work consists of drill.

GALLER: One technique that we used in our Ford Foundation project was to get the deans of the various schools to assign full professors to be assistants and recitation

instructors in some of the courses for our sophomores. They suffered, but they learned.

DAVIDSON: Bernie, you've agreed that there should be such a course. Now, would you add to that and state what you think the content and goals of such a course should be?

GALLER: I would go along largely with the things that Joe said earlier--the organization of problems, algorithms, data representation, subroutine hierarchy, and the hierarchy of problem solutions.

MILLS: What we should be teaching them is actually doing a job on the computer. I believe that anyone learning anything about computing should get on the machine and actually do it. It disturbs me that we discuss lofty goals, like the teaching of concepts, and carefully avoid talking about programming.

SHADER: There's a definition used by Bill Dorn that I like. He says you should teach them exactly as much programming as they need to do the problem and no more. I know you don't like that, Roger, but he's not trying to teach trained programmers.

FINERMAN: The point is that you may not have to start doing that in high school.

Let me mention just in passing a conference that was held at Stony Brook back in June. It started out to discuss the subject of graduate and related research programs in computing science, but because it became much broader than that in scope, it will be published under the title "University Education in Computing." Although the conference was devoted to graduate academic programs, the most controversial topic of discussion proved to be the undergraduate program. More specifically, the discussion centered on whether there should be an undergraduate program leading to a degree in computing science. Representatives of industry and universities examined the needs of

industry and graduate schools (of computer science). One conclusion reached by many was that there should be no undergraduate degree in computing science. One of the arguments advanced was that undergraduate specialization would leave relatively little for study at the graduate level. One has to recognize that the field has a certain degree of narrowness at the present time. True, this narrowness may disappear in the future, but today there is a limited range of subject matter. Also, representatives from industry agreed that they would not want to hire someone who had both a Bachelor's and a Master's degree in computing science. This educational background would be too narrow for industry's needs, and they would prefer a broader scope of course material.

SANDERS: Perhaps even a Bachelor's degree is too much.

FINERMAN: The feeling expressed by many was that too much specialization at the undergraduate level was unwise. The argument was that at the undergraduate level students should be exposed to fundamentals and not to specialized courses. The person who wants to specialize can do it after his Bachelor's degree, either through experience in industry or by further study in the graduate school.

The same argument (if it is valid) can be extended to computing science programs in two-year colleges. There seems to be a general trend in the colleges to offer an Associate degree in computing science. Hamblen's survey, recently published, shows that there are an enormous number of two-year programs in computing science. These programs allow for very little in the way of a broad range of fundamental material; much of the student's time is spent in specialization.

Now we seem to be discussing inflicting the specialization at the high school level. It seems to me that, at some point, we must decide upon the purpose of these programs and

how they serve the goals of formal education. Do we want to give students a broad educational background in fundamentals, be they in liberal arts, music, mathematics, physics, and the like? Can we, at the same time, give students the capability of solving problems? Do we want to give them, in addition to a broad liberal education, specialized knowledge of coding, programming, and problem-solving? Should this specialization be initiated in high school, in the two-year college, or even the four-year college? Do we really want to train kids specifically to be computing scientists or computing engineers?

I hate to see a trend toward teaching details of programming, as such, in the high school. It makes more sense to imbed the computing concept in a mathematics course, say, where certain fundamental concepts could be demonstrated on a scope display. I think it is better at the high school level to use the computer as a tool in fundamental courses to make information available instantly. More detailed knowledge of how you make that information instantly available should be deferred until the college level.

MILLS: That may be true. I myself was hooked on mathematics in high school. I fell into programming because at that time a mathematician had no other place to go. But it seems to me that it is among the high school population that we are going to get our programmers for the next ten years.

FINERMAN: But there are so many more important things to give students in high school.

MILLS: But why not give them a chance? In our schools today we expect a student at the end of his freshman year to have picked his vocation. There is a heavy emphasis in our high schools on having the students decide right then what courses they are going to be taking in college.

FINERMAN: It is a sign of our times that fewer and fewer of the students coming into college know what they

want to major in. It is also a sign of the times that much specialization previously available at the college undergraduate level is now deferred until the graduate program. At the undergraduate level we should give fundamental courses in mathematics, physics, and computing science.

ATCHISON: But we must recognize that most of our discussion now is only theoretical. On the basis of Hamblen's report, whether or not we like the idea of programs in computing science, we have them in very large numbers. There is no doubt that we are going to have more such programs as time goes by.

SANDERS: I can testify as an employer of such people that we don't want them. They are useless because they don't know what to program.

MILLS: Do you solve that problem by putting the course content in the high school and making it redundant in the junior college?

FINERMAN: The blunt fact is that the two-year colleges are finding that this is a commodity that can be sold. They are, in a sense, running a business like anyone else, and they constantly seek a salable product. Two-year colleges believe that an Associate degree in "computing science" is a salable product. I believe this does a disservice to the student, but apparently colleges do not lack students in such programs.

SHADER: All of us here work for leading organizations--the most sophisticated universities, the most sophisticated manufacturers, the most sophisticated industries. The fact remains that there are many other industries that are desperate for people with the type of training that Finerman was talking about. There is a tremendous demand (and it is growing) for people who have the terminal training that can be given to the 40 percent in the high schools and is now

being given to students in the junior colleges. It is useful, I think, to satisfy the demand for computer operators or low-level programmers. You may have to neglect a few of the good students, but you will be doing a service to a much larger group.

FINERMAN: The need that Shader expresses for people with the high school or junior college terminal training is mainly a need on the part of some industries for low-cost labor. Many companies have grown up with a tradition of punched-card tabulating methods with which they got along fine (or thought they did) with low-cost people. These companies would like that situation to continue when they switch to computers, since they regard these simply as extensions of tabulating devices. Other users who regard the computer as a more sophisticated tool, and who do not try to parallel by computers what was done with manual or punched-card methods, realize that they cannot survive with low-paid help who have the kind of training to which Mel referred. Those who use computers properly do not, as a rule, hire high school graduates or those with only a two-year degree. The only companies who hire the two-year graduates (as programmers or junior programmers) are those who do not use the computer properly. It is these firms who may abandon the computer. And when they abandon it, they complain bitterly that it is no good and hasn't performed as expected. These firms never realize that their failure derived from personnel without sufficient background to plan for a computer installation properly. The computer is a hard taskmaster, and it will not be satisfied with low-cost, untrained personnel.

I don't think it is proper for our educational institutions to go along with a fiction that modern technology is satisfied with poorly trained people, under-achievers, or whatever you want to call them. This does not imply, in any way, that there is no place for the two-year college. There is; however, I don't think that it is possible for the

two-year college to produce a graduate whom you could call a programmer or even a junior programmer (whatever that is). I feel that is simply a fraud. It is not the function of educational institutions simply to train people for industry, and even in doing so, to do it improperly. It is improper to foster the fiction that two-year graduates will be able to program sophisticated administrative applications. The computer devours untrained people and the companies who hired them.

WHITE: I can reinforce what Aaron just said by relating something that happened here not too long ago when Harbor Junior College conducted a meeting of representatives of the large computer users in the aerospace industry and others. Those who attended were in personnel, and about 30 of them came to the meeting. The object was to determine what the junior college should be offering. The junior college was asking in effect, "What kind of computer science curriculum should we set up to furnish you the type of person you want?" After a day's discussion, they decided that there is no such program and that industry could not use the output of the two-year college. People with a background of only two years in college just do not have the education, not to mention the technical training, that industry wants.

WEIZENBAUM: Where do they get the people who do the job that they want?

WHITE: They hire exclusively graduates of four-year colleges.

WEIZENBAUM: I was asking a leading question, of course; I anticipated the answer I received. I think an error is being made but not on the part of the educational institutions unless we have some conscious frauds operating, and I know of no such instance. The error is being made on the part of the employers. They must be deluding themselves for

one reason or another into thinking that the subject matter with which they deal is irrelevant or trivial. They must believe that just because someone has learned how to manipulate things that he can be hired and thereby become an aerodynamic designer. I can postulate Ph.D. mathematicians (and cite examples) who understand about solving differential equations, and they have heard about tensors and vectors and things like that, but if you sent them into an airframe company, they would be totally useless. They would not know the substantive material that is being dealt with in the aircraft company.

Why should one expect a high school graduate, who has been made into a skilled programmer (in the same sense that someone else has been made into a skilled arithmetician), to walk into an aircraft plant and know what he is doing?

SHADER: Let me ask you this question, Aaron. If the two-year colleges are not capable of turning out people who are useful to industry, what does happen to their graduates? They don't all go on to the four-year schools.

FINERMAN: I was trained as an engineer, and engineering went through this experience some years ago when two-year colleges felt that they could turn out engineers. There is a long history of a time in which two-year colleges were actually trying to compete with Bachelor-degree institutions in producing engineers. Finally, the two-year colleges realized what it was that they could turn out--namely, engineering aides. The engineering aide, as the name implies, assists the engineer, perhaps by doing drafting chores. There are many areas for which an engineering aide is well trained. At my computing center, I find that the graduates of the two-year college make wonderful operators and operator supervisors.

MILLS: So they should be turning out computing aides.

FINERMAN: That's fine if the two-year colleges recognize that that is what they should be doing. The trouble is that right now they don't recognize that this is their function. Perhaps as a result of industry pressure, they claim to be turning out programmers or, at a minimum, junior programmers.

GRUENBERGER: It seems to me this discussion has gotten thoroughly out of control. This all started with some questions from Davidson. Charlie, would you try to get this discussion back on its track again?

DAVIDSON: I'll put it in the form of a new question. Is there a course that should be taught in high school in computing, and if so, where? I haven't heard any answers to these two questions.

GALLER: I'm not sure what it means to have a course about computing, and that's what we have been trying to find out. At a minimum it means a course that wasn't there last year in which computers play some role.

DAVIDSON: Our discussion before lunch and the content of the film, it seems to me, involved the general education of high school teachers about computers. I think the object was to enable the teachers to integrate concepts about computing with the subjects they are now teaching.

GRUENBERGER: If you're referring to the film, the goal there was quite specific. We were told that the school board was going to put computers into some of the schools, and these teachers were there to learn how to teach computing.

DAVIDSON: That I didn't get from seeing the film; in fact, I got quite the opposite impression.

FINERMAN: The hardware in the high school could still be imbedded in a math laboratory and would not have to result in a separate course devoted to computing.

DAVIDSON: Or to various areas of application.

WEIZENBAUM: Let me take the bull by the horns, Charlie, and answer your question. You ask whether there should be courses in high school devoted to computing. I say, yes, there should be; and I feel that in five to ten years they will be common. The only question I see left is what influence can we exert as to how it should be done and for what purpose? The way it is going to develop at least initially, I feel, is very much like high school chemistry or physics. In effect, it will be a computer laboratory. I don't think that initially schools will attempt to teach geometry or algebra using the computer. It will be similar to the high school chemistry or physics laboratory and may fall under the rubric of general science.

FINERMAN: Do you think this will be similar to a problem-solving laboratory?

WEIZENBAUM: I don't know, but I am sure that is not a bad idea.

I think that it will involve something lke a PDP-8S small computer or perhaps a timesharing terminal, and the teacher will stand in front of the class and explain how the computer works. This explanation will be on a level appropriate to the kind of student we are talking about. After some explanation there will be problems to solve, and the whole thing will be done mainly by an illustration. I would guess that there would be heavy weight on numerical problems, but I would hope that there will be non-numerical problems as well.

The biggest benefit will come from the side effects. In the university atmosphere where computers have existed for quite a long time, the side effect has been the integration of interdisciplinary work, and this side effect is probably more valuable than the presence of the computer itself. In a similar way in the high school atmosphere the

side effect will be the introduction of the computer to all the teachers in the school so that they will indeed use it as their metaphor. I think this side effect will be more important than the direct effect of the computer and that after ten years or so, we'll get some good kids out of this system.

DAVIDSON: I wonder why you are so optimistic when we have not seen that side effect take place in the colleges after ten years.

WEIZENBAUM: I think the answer is that we are still a young field. It isn't a sequential phenomena that must take place first in the colleges and then in the high schools. It has not yet permeated our culture, that's all. It is now beginning to do so. It is what happens with all forms of maturation. The glamorous, the colorful, and the spectacular take the spotlight for awhile but eventually recede into the background where they take their rightful place. We are now just beginning to see this receding into the background taking place with computers and it is not at all surprising that it has not yet occurred in the colleges.

As I understand what you are doing in Wisconsin, you have already somewhat reached this point in that the computer has receded into the background.

DAVIDSON: Within the College of Engineering this is beginning to be true, but it is certainly not true with the other departments of the university.

WEIZENBAUM: It will take time.

GRUENBERGER: It hasn't even been codified yet.

SHADER: We do have some statistics to add in here. Just ten years ago the number of computers in universities was about 20. And today, it is something over 500. There are over 200 computers now installed in high schools.

FINERMAN: Joe says that this will come about, but I doubt it. We live in a pragmatic world, and I don't think the chairman of the physics department will want to give up part of his physics courses in favor of a course in computing.

KREHBIEL: Might not computing in the high schools follow the pattern that typing has? We have two types of students who take typing. There is the honor student who takes it, perhaps, in summer so that he will have an additional skill that he can use for his own purposes. Then there is the other type of student who takes a full year of typing for vocational purposes.

For the bright student, computing need not be as detailed in its introduction as it would be for the vocational student. A summer course early enough in his career, followed by encouragement from the teachers of his other courses, would nudge him along all the way through so that he probably wouldn't need any more formal training.

SANDERS: Why are we pushing the lower-class guy into programming at all? It seems to me that we are making a promise to him that we can't fulfill.

WEIZENBAUM: Are we making him a promise when we give him a high school physics course? It is simply part of the culture.

Two hundred years ago in England, educated people could not calculate the making of change. It wasn't required, and it was considered rather difficult. Today everyone learns how to do that calculation; in fact, when I go to London I find people quickly calculating the correct change in their funny money, and I can't keep up with it.

FINERMAN: You don't give the poor high school student the course in computing.

WEIZENBAUM: I disagree. You give even the poor student all the basic courses in arithmetic, English,...

FINERMAN: But you don't give him physics.

WEIZENBAUM: But you probably will. Things happen much more quickly today than they used to. That is exactly my point: within ten years everyone will have access to computers in high school. This won't happen because anyone is promising anything to industry either. It will occur simply because it is part of our culture. If driver education has to drop out and make room for it, it will.

MILLS: One of the beauties of the computer, it seems to me, is that you can do problems that will illustrate points in many other subjects. The availability now of remote terminals allows us to do this job even better.

Computers can also relieve the dog work and thereby make themselves useful. I remember how much I liked trigonometry until I reached those miserable logarithms and became mired down in the arithmetic. Think what a useful tool the remote terminal would be in that atmosphere. By taking the dog work out of other subjects, computers can lead us back to the educational process that we should be spending our time on.

DAVIDSON: You're putting the emphasis on the computer as an adjunct in other subject areas and not as a subject in itself.

MILLS: That's right. But to use it, you have to learn all about it.

DAVIDSON: Then my original question is still here. Is there a place for a course in computing in the high school?

WEIZENBAUM: I thought I answered that.

MILLS: I will answer it too: yes. I just got there by different reasoning. If I tell a student to put his chemistry problem on the computer, he isn't going to be able to do it unless he has learned something about computing.

DAVIDSON: But that's not a course in computing.

MILLS: You have to do first things first.

DAVIDSON: But that isn't a course; that's just one to five hours of training.

WEIZENBAUM: But I say again, I thought I answered that question. I think there will be a small computer or terminal, the teacher will introduce it by examples, and the students will gradually learn about computing. I don't have a crystal ball, but it is my conviction that this is how it is going to come about.

MILLS: It doesn't have to be a full course, but there should be a full course available for the student who is interested enough to take it.

ATCHISON: I'll support what Joe is saying. He is putting it on a theoretical basis. I'll put it on a practical or pragmatic basis. It is happening right now. That's the point that Mel was trying to convey before. With 200 computers already installed in high schools, it is now going on. It is a lovely academic discussion as to whether or not there should be such a thing, but it is only academic.

WEIZENBAUM: I mentioned before that many of the kids coming to MIT now already have a considerable knowledge of computing. I'll agree that MIT is not typical, but it shows that somewhere out there something is happening.

ATCHISON: As an example, we heard last weekend that every high school in Philadelphia, except one or two, is now getting computing. In these schools it will affect every mathematics student. Although it is only in the mathematics department at the present, it will certainly spread, and the same phenomena is taking place across the country. I am also happy to report that this isn't restricted to our country. At a recent IFIP meeting that I

attended, I was hearing reports of the same thing happening in other countries at just about the same speed.

SHADER: In the other countries it's not happening on such a broad scale, but it is happening in the same way.

GRUENBERGER: It has been present for ten years now in some of the better high schools.

KREHBIEL: Charlie, are you raising the question to discover if it should be subject oriented or non-subject oriented, or whether it is to be universal or only for college preparatory students? Or what?

DAVIDSON: I was asking whether it should be subject oriented, in the sense of being an adjunct to other subjects, or whether it should be a principal subject, with other subjects being considered as areas in which to apply the computer. But more important to me than simply observing that it is happening and we should get in on it, I am still fighting with the other question: namely, should it be happening? Do we honestly feel that we should be giving this level of computer education in the high schools, and if so, toward what goal?

WEIZENBAUM: I say yes. I tried to answer his question simply and straightforwardly. I think it should happen.

SHADER: But it is not happening just that way. For example, in Philadelphia it is mathematics that is being taught with the aid of the computer. It is not computing being taught in and of itself. That idea is now receiving widespread acceptance in the mathematics community.

GRUENBERGER: You don't have any worries then, Mel, because no matter where you introduce it in the schools, the kids will learn it very fast anyway.

I am with Joe. It is happening whether we like it or not, but I like it. I think it can and should be a separate course.

I'll raise the question, however, that we continually meet from the school board; namely, what has to go? It is a particularly acute question in California because here every hour is loaded, and they like to think that each of those courses is important. They reason that a course in computing will push aside some other course, and they want to know which one I think it is going to be. The only glib solution that I can offer is that it might be imbedded inside a course in trigonometry where, as Roger indicated, in relieving the dog work it makes room for itself and pays its own way. In other words, the student regains the time he spends learning the computer in the trig course. (I should point out that the school administrators I've tried that on don't like that solution at all.)

ATCHISON: If we can comprehend that for good or bad this movement into the secondary schools is happening now, then we should focus our attention onto what we can do to make it effective.

ARMERDING: I was taught the slide rule in high school as part of some other course, and I don't even remember what course it was--probably one of the science courses. I assert that I can take anyone you can name (who is reasonably intelligent) down the hall, plunk him in front of a JOSS console, and teach him how to use JOSS in about the same time it took me to use the slide rule. Why does it have to be a separate course in the high school? Why do you have to fight with the school board? Why not just slip it in at whatever point is natural for the student to learn how to use this beast?

MILLS: My point was that there had to be a separate course for the student who wanted to go on.

ARMERDING: Fine. That corresponds to a real course in advanced slide rule techniques.

GALLER: But if that is all you want to teach, you shouldn't have a separate course. If that is all you need, you can have a lecture some evening and invite them to come, and they will all be there.

ANDREE: What else is there?

GALLER: Well, I thought we listed a little while ago all the other things you could teach about computing. Should we list them all again?

FINERMAN: If you assume that teaching someone how to operate a JOSS console or how to use JOSS language is equivalent to computing, you are making a dreadful mistake. We have mentioned several times that it is fundamental concepts that should be taught. If you equate this with an advanced course in the use of the slide rule, then you have missed the whole point. This is not to say that a JOSS terminal or a standard computer could not be part of the learning process, but I hope you pay some attention to the prime requirement.

MILLS: I think you can do both. The analogy with the slide rule breaks down all over. The slide rule isn't a computer; it isn't very expensive, and you aren't keeping anyone else from its use while you use your own personal tool, and so on.

There is a responsibility that comes along with computing. If irresponsibility is given to the student, it is terribly difficult to get rid of it, and it is one of the chief problems we have in industry. The course that Bernie was talking about would be designed to teach them the proper responsibility that goes with the computer.

ANDREE: I think it is highly desirable in a high school to teach a student how to write a clear English sentence. I think also that it is highly desirable to teach a student in high school how to write clear communication to a computer. I have no quarrel with anyone who

wants to go beyond teaching the use of a clear English sentence to the beauties of Beowulf, nor do I have a quarrel with someone who wants to go beyond the teaching of basic communications with a computer--provided that the person who wants to do this knows something about it.

WHITE: How do you teach further aspects of computing outside of the problem-solving context? You have to give them something they can work with.

GRUENBERGER: If I get what you are driving at, Bob, didn't we make one attempt at that by showing the teachers at our workshop examples of computer programs written by masters?

WEIZENBAUM: I'm not sure I can give the answer to that question, but I can give an attempt at an answer. I mentioned earlier that we have a computer program at Tech (for which I am responsible) to teach programming. This operates on a timesharing console. I mentioned before that it is not designed to teach a language. It is a course to teach programming. I can believe that one can learn the JOSS language in an hour or two. Theoretically, the language that we use in this terminal system should be learned in an hour or two, but it takes many hours. Why is this? The reason is that we are trying to teach ideas that will be transferable and that have little to do with this particular language. The course is designed for college students, but I don't think that it will take very long for it to descend.

Let me give you an example. One of the things we want to teach is the difference between an equation and an assignment statement. Another notion is the idea of an identifier. What is the difference between the notion of a variable as we use it in computing and the concept of a variable as it is used in mathematics? Most particularly we are trying to teach the ideas of subroutines, loops, and iterations.

We are trying to teach what I can refer to in the context of this group as a FOR statement (but notice that I couldn't use that notation in the context of another group). We provide motivation (and I think this comes close to answering your question) by showing the student how to use something. The language that he is being taught to use, although all of it is in the computer, is locked off from the student at various levels. If a student in his third lesson tries to use some advanced part of the language, he will be told that it is an illegal statement. But in his fifth lesson it might become legal. At the earlier stage we have locked it out from him.

We give the student a problem and ask him to do it with the tools he has learned so far. When he has done it, we say to him "Wasn't that rather awkward; wouldn't it be more convenient if we would do it this way?" We might, for example, point out to him that it would be handy to enter a subroutine with parameters. The whole idea of parameter communication is one of the basic things we are showing him. We then unlock that part of the language and allow him to use it. Each time, I can promise you, he sees the need for a mechanism before he sees or can use the mechanism. When he gets through, we hope he will have a good idea of what some aspects of computing are all about.

We have very carefully chosen students who are truly novices to try this course. Such people are hard to find at Tech because most of the incoming student body already knows a great deal. I claim that when they are through with this course at the end of the semester, I will be able to go to them and show them in two or three lectures the system that they have been using and how it works, and they will understand it. If what I claim is true, I think it is important.

WHITE: If we offer a class in computing, it has to be fairly early, or it won't be of any use to the student

in the classes to which he wants to apply it. But if we want to offer a general computing course in the high school (for which I have just concluded that it has to be fairly early in the high school curriculum) we don't want to offer it just to the mathematics-oriented student. How are you going to phrase the course so it will be understandable to the student who has a bent toward the social sciences as well as the kid who is mathematically oriented? For these two types of students the course would have to be phrased in entirely different terms.

FINERMAN: I am not sure what you say is so, and I feel that the problem is more complicated than you suggest. I recently met with people in the social sciences who want us to offer their students a computing course. I spent an hour describing the introductory course we now offer; their reaction was that it was a wonderful course but not meant for their students. After more discussion they felt that it might be good for their students but only their better students. In effect, they wanted two courses: one involving the abstract concepts (in the present course) for their top-level students and a different course for their not-so-good students who are only interested in analyzing surveys and similar problems.

WEIZENBAUM: But there is nothing simple about taking a population survey. Computing in the behavioral sciences is every bit as difficult as it is everywhere else and maybe more so.

FINERMAN: But that is my whole point. Computing is every bit as difficult in the behavioral sciences, in the management sciences, in medicine, in industry, and in all areas utilizing the computer. It is not simply a tool and cannot be treated as one.

KREHBIEL: I hate to see the years 12-16 passed over here without some mention because those are the years during which education is the most efficient.

CANNON: I think it would be well eventually if we could integrate the computer into nearly all courses. But in the interim might it not be well to have a lab-type course offered for those students who are looking for something that they regard as lighter and can look on as fun? I would rather have it on that basis than as a tough, required course because I don't have that much confidence in the way it would be taught. As a rigorous "solid" course, there isn't room for it in the high school curriculum. As a useful diversion for those who are interested, there is plenty of room.

EVANS: The conclusion we must reach is that although it would be desirable to have the computer around influencing all courses in various application areas, we will not have that for another generation, and hence we're left with the concept of a separate course devoted to the computer and computing. The experience we have had in the colleges makes this clear. Even today we cannot rely upon teachers in the other disciplines inserting the computer and its concepts into their teaching. We still must have the computer and courses in computing as separate entities in college. If it is not taught as a separate course, it is likely to be taught badly. Notice that we have separate English courses in our high schools. But we learn English, if at all, through its use, mainly in other courses.

DAVIDSON: There is a corollary point that should be mentioned before we get off the subject. We have talked of the hope of having the students diffuse the knowledge of computing, if we get them started and give them access to the machine. The key assumption that we make there is that the computer is available to the student relatively easily most of the time. It is my conviction, and I would hope that those here would agree with me, that we cannot afford to fool around with project numbers, instructor authorization by course numbers, and satisfying government auditors--

that the computer must be as free as the library is in the high school (and as free as we hope to make it some day in the colleges).

GRUENBERGER: Those things aren't completely inconsistent. At Valley State College we give them a magic number in each class, and from then on they sort of have their license to use their machine all they like.

EVANS: You don't believe, do you, that the amount of available computing time is unbounded?

DAVIDSON: No, of course not. Some limitations must be set up, whether it be five seconds or five minutes of computer use without question. But the analogy to the library is quite strong. In a library a student can get a book (most books) and use it freely. For certain books he is restricted to the reserve shelf; he must consult reference books only in the library, and so on. You can set up a system of rules like this which essentially makes what the student needs freely available to him.

ANDREE: I have been invited to go to a high school next week and address all the teachers in that high school on the subject of computing. Apparently the request for this has come from at least some of the teachers.

I was delighted to get this request, but I am beginning to get some qualms about what one says to *all* the teachers in a high school about computers. What do you suggest?

GALLER: I would suggest that you try the topic: "What Is It That a Computer Cannot Do?" On that framework alone you could prepare a nice talk.

GRUENBERGER: Secondly, avoid the *Life* magazine approach: "The Computers Are Taking Over."

SANDERS: Are we going to list for him all the slogans that Joe has warned us not to use?

KREHBIEL: Thirdly, I suggest that you could say to them almost verbatim what Professor Davidson just said regarding making computing power freely available to students in a way analogous to the library.

SANDERS: But they don't know where to go for that access to computing power.

ANDREE: This particular high school is within half a city block of the installation that is going to have a 360 Model 40 with time-sharing, and the school fully expects to have a terminal.

GRUENBERGER: Don made a point awhile back that we might pass on to the school. If they get involved in computing, they will reach a point very quickly where the smartest thing they can do is to get out of the way of the students.

CANNON: You might also reassure the teachers that they will soon get over the trauma and fear they may now have that computers are not easy to get used to.

KREHBIEL: They should be told that their students should be encouraged wherever possible to use the computer to do what homework problems they can and to exercise their ingenuity. The teachers should be told they should accept these solutions, and if all this comes about, they will learn very quickly where the computer applies to their subjects.

GRUENBERGER: But you are going about it just backwards, Don. The thing to do is have these teachers warn the students that they should never under any circumstances be caught doing any of their homework on the computer. That's the way to guarantee that they will stay up all night doing it.

KREHBIEL: Yes, it works that way. Seriously, though, the important thing is that the teachers should insist on good solutions. That implies that they must learn the difference between a good and bad solution.

GALLER: Krehbiel said something that I would like to comment on. He said that the teachers should insist on good solutions rather than brute force solutions. I think that is something that should come much later. If you try to start out with that approach, you will kill the whole idea.

KREHBIEL: I'll substitute "encourage" for "insist." Actually I don't mind accepting bad solutions as long as I can turn them around to the student and encourage him toward a better solution. The important thing is to start an awareness on the part of the student of what is a bad solution and what is a good solution.

If I were in your position, Dick, I would, perhaps, feel some qualms, but not too many. I am not familiar with the subject matter in other fields. I assume, then, that the teachers of the other subjects will have to create the interface between their subject and the computer.

ANDREE: I figure that with a group like this there should be several teachers who have had some experience that I haven't had that I can capitalize on.

GALLER: Something you might try is that after you have talked for awhile, you can then throw the floor open to them and have them ask you whether certain things can be done with the computer.

GRUENBERGER: It's been my experience in a similar situation that you can't shut that kind of stuff off. It is inevitable that they will ask questions of that nature.

WEIZENBAUM: Yes, and I object to that sort of thing. This whole business of "Can the computer do...?" is a terrible thing. In the sense that those teachers will frame the question, the computer can't do anything. We should all know that a person using the computer as a tool interacts with it to do something. Perhaps you should encourage them to

reframe such questions in terms of what the current limitations of problem solving are.

In two hours you will be lucky if you can make two good points. I would encourage you in two hours to try to make just two points and make them very well.

GALLER: And the two points are that you have to be able to recognize the problem and define it and that you must organize an intelligent attack on it.

I think the subject of language translation is a fine example. To say that the computer cannot do language translation is a meaningless statement. When you explain what the limitations of our current efforts in language translation mean, then you have explained something.

WEIZENBAUM: I'm reminded of a PTA meeting I attended at which a psychiatrist talked. One of his subjects was the anxiety that boys have about showing their emotions in public. He used, as an example, the experience of a boy going away to camp and described how most parents don't prepare a boy for the anxieties and emotions he's going to experience during his first time away from home. It was evident to me while he was talking, and it was verified later by the questions the parents asked, that they were concentrating all their attention on what they should tell their son when he goes away to camp, and that wasn't his point at all. We have the same sort of phenomenon taking place here. People choose an example to illustrate a point, and we get to discussing the example, and the point gets lost.

A talk such as the one you are going to give, Dick, is very important, and I recommend strongly that you pick one or two points and make those points as strongly, as honestly, and as deeply as you can. It would be far better to leave them with one or two points that you have made

deeply than to try to cover thirty or forty points superficially. If you had a half-year to make your points, it might be different.

ANDREE: I'd just as soon not have half a year.

EVANS: But you ought to be aware that you might get the half year.

SHADER: It might be useful to show them the film we just saw.

GRUENBERGER: I'd like to propose a new topic, which does not appear on our printed agenda, and I will read it to you as I have written it:

> 12) What mechanism do we have for the interchange of news of current activities in our colleges? Formal reports go to the *Communications*, of course, but how do we exchange information on informal activities?

I think we have the same situation now that we had back in 1953 that led to the formation of the newsletter, *Computing News*. There is much information that would be useful to people that is not in the proper form to be printed in our technical journals and is dropping through the cracks. The example we had before of the study at Michigan that Bernie mentioned just appalled me. This has been out for four years, and I didn't even know it existed. We have nine colleges and universities represented here, and I don't know what's going on at most of them. I have a pretty good idea of what's going on at Wisconsin because I talked to Charlie, and I hear from time to time from Bernie of things going on there, but I really don't know what activities are current at most of the schools in the country.

SANDERS: I don't think you'd have time to read it if it were sent to you.

GRUENBERGER: Perhaps there is need for some other publication, although we have too many of those already; perhaps what we need is a new department in the *Communications*.

ANDREE: The people who are doing things don't have time to write it up.

FINERMAN: That's true, no matter what system you get up. The important thing is to have the inputs created. Most people just don't have time to create those inputs.

GALLER: What sort of information is it that you want?

GRUENBERGER: I would like very much to have known about that study at Michigan. I would like also to have heard about the language that Andree told me last night they have been developing at Oklahoma, which sounds very interesting. I would like to have some sort of informal interim report on the programming course at MIT that Joe was telling me about. Sooner or later I will hear about these things in the form of formal reports, but by the time it gets to that stage, it's far too late to be useful to me.

ANDREE: Something you should know about is the work that Ginsburg has been doing here in California on context-free languages.

GRUENBERGER: Are you fellows telling me that all this information is reported, and I'm the only one who doesn't read?

KREHBIEL: All too often someone asks me when I'm going to document what I'm doing, and I ask them if they want it running Monday morning or if they want the documentation Monday morning; they can take their pick.

WEIZENBAUM: I choose the documentation. If I can get the documentation by this Monday, I am sure you'll have the program running by the following Monday.

I once had a fellow visit me from a German newspaper. He was the science editor for the paper and had, I believe,

a Ph.D. in physics. He was a professional journalist with a good background in science. I was busy at the time, so I arranged with him that I would continue doing my work and from time to time explain to him what I was doing. He spent an entire morning with me. A couple of months later he was courteous enough to send me what he had written in advance of publication. It was the best piece of reporting that I had ever seen.

We need a high-quality journalist who thinks of himself primarily as such and not as a scientist. Such a man hired by ACM or AFIPS could go around and nose out the news.

If this were established, not necessarily with one man but perhaps a small staff, there might be an interesting byproduct. We do not have in our field a tradition of criticism. We don't have any grand old men whose competence we would all acknowledge and whose criticism we would welcome even when they knock us. We do have *Computing Reviews* and, generally speaking, what appears in there is very good. What we lack is the senior authorities who are no longer actively producing but who could still comment on the field. Imagine von Neumann retiring from active work and devoting some of his time to criticizing.

My point is that a journalist--a professional journalist--would not wait for news to arrive at his desk but would go out and find it. That's what a journalist does.

EVANS: There's another problem too. Whatever is formally published in our field is already late. For the kind of information you're talking about to be useful you have to be aware of it a long time before it is formally published.

GRUENBERGER: Yes, there's a principle that Joe enunciated two years ago. He pointed out that the only intelligent form of information retrieval is the telephone. When you want to know something, you pick up the phone and

in no more than three levels of addressing, you find out what it is that you want.

MILLS: Since we have representatives here from both AFIPS and ACM, why don't we simply make a recommendation to them, jointly or severally, to look into this idea that Joe has brought up. They could use the pages of the *Communications* or some other journal as their avenue of communication. I would think that the professional societies would be delighted to have this done if there were a way to do it, and it shouldn't be restricted to news of the work in the universities.

DAVIDSON: ACM already proposes to establish a new publication which will be more of a newspaper, containing news, notices, and other items that are now in the *Communications*, along the lines of quick dissemination of information rather than scholarly information.

GRUENBERGER: In other words, the sort of stuff that the *Communications* was originally set up to do.

Joe, let me get your opinion on a couple of matters here. Would the man you picture have to be an old respected man in the field, and would he have to travel? Could it not be done by a young, eager journalist type who stays in one place and uses the telephone a lot?

WEIZENBAUM: You have two things mixed up. The old man in the field I was referring to is for the role of critic. The journalism job could undoubtedly be done by a young person and would perhaps be done better. What I picture is somebody who did his undergraduate work at a recognized school of journalism and then did a year of graduate work in computing with emphasis on electrical engineering.

GALLER: I do think he would have to travel.

SANDERS: Yes, he would have to come to us and watch us at work.

GRUENBERGER: That's a very interesting point. If we had such a person and he did travel, when he reached the Pacific Northwest, he could concentrate everything he wants to know in one visit to Sanders. In the same way, I would guess his target in the Boston area would be Joe, and I would imagine that we could find a similar target in every major city. There is always someone who is a source of information and knows everything that's going on.

GALLER: I'm still curious to know what information it is that you would expect this man to dig out.

WEIZENBAUM: For example, if you had been around Project MAC at the time this particular journalist had written his article, you would have received good news. The news was well-written, interesting, and informative. It was a short package so that you would at least know the system existed, and you would have enough information to be able to write a letter or make a phone call to find out more. As Fred pointed out, the problem is not so much that we don't have time to read the technical articles, but we just don't know what's going on.

GRUENBERGER: There's always the phenomenon that when you plunge into something new and novel, it may be that three other groups are doing it at the same time, and none of them discover the existence of the other until a year later. I'm suggesting that we should have a mechanism to short cut that.

DAVIDSON: I don't think we have answered Bernie's question. He asked what kind of information--and this means what field or what topic, with what currency, and aimed at what audience--are you looking for that you feel does not now exist or is buried too deeply to find readily.

GRUENBERGER: That's a fair statement, but remember, I qualified it by restricting it to college activity. I want

to know as much about university X as I happen to know about Michigan through you, Bernie. I get to know what is going on at Wisconsin because I get to see Charlie about once a year, and I say to him, "What's going on at Wisconsin?" and he tells me. And 90 percent of what he tells me is never in print anywhere. The various publications tell me that Wisconsin has ordered a Burroughs 8500. That's just a bare-boned fact. When I see Charlie, that's worth a couple of hours of discussion; what are the implications of Wisconsin ordering a B8500? What I get from a discussion with Charlie is a feel for the computing milieu that is going on at the Wisconsin campus. I'd like to have that same feel for many other campuses.

ATCHISON: I'd like to have the sort of feel they're describing in relation to what is now going on with CAI at the various campuses.

GRUENBERGER: Someone mentioned before that there were four or five hundred institutions of higher learning that now had computers. I know what the bulk of those are. They're small schools that are teaching essentially punched-card data processing, using perhaps a small computer.

DAVIDSON: I think you're a few years out of date, Fred.

GRUENBERGER: Like so much I'm out of date. I've seen some of these schools; we have a few of them around here. I see the publications--articles and books--they're producing, and I'd be willing to lay bets I'm right.

GALLER: I'm afraid I have to back up Fred. I visited a large midwestern university recently--one having over 18,000 students. The person who showed me around was one of my own former students, and I'm not too proud of what he's doing. Specifically he is concentrating on raw machine language and staying there far too long.

GRUENBERGER: I just reviewed a book for *Computing Reviews* with a title like *Fundamentals of Data Processing*, which I would assume meant that somewhere along the line it would involve computers. The book was full of plug board diagrams, descriptions of collators, reproducers, offset gang punching--the sort of stuff that we abandoned under the heading of computing or data processing over 15 years ago. The only thing this guy left out of the book was a discussion of spirit duplicators. I'm fond of all those machines, but I wouldn't offer them to innocent students now under the heading of "Fundamentals of Data Processing." I think that's an out-and-out fraud. A student who takes a course with a title of Data Processing has a right to expect that he is near the year 1968, and he deserves a small peek at 1969. If he knew how he was being cheated, he could complain somewhere, but these things get by because the students don't know. They think they are learning computing and/or data processing.

ATCHISON: The group that was formed of directors of university computer centers had as one of its goals the dissemination of the kind of information we've been talking about. It hasn't worked out too well.

DAVIDSON: That group was not organized primarily to deal with instructional problems, but rather with operating details of university computing centers.

GRUENBERGER: One thing that could be very useful would be to systematically collect the sort of statistics we were groping for this morning--how many students take computing in relation to the whole campus, etc.

FINERMAN: It would be a monumental task to gather that sort of information.

GRUENBERGER: The reporter we have in mind doesn't have to generate these surveys but simply report on them when someone else makes them.

CANNON: The whole idea of a professional journalist seems like an excellent and rather exciting approach. Is there something we can do to have it implemented?

SHADER: I will certainly transmit the idea to the AFIPS board.

FINERMAN: ACM is looking into such an approach at the present. We expect to take formal action very shortly.

GRUENBERGER: Maybe it's just part of a much larger problem. Maybe we're only commenting that the *Communications* is now mature enough to warrant having a professional journalist on its staff.

[Agenda topic #7: what should be done about teaching computing to adults *not* in school?]

FINERMAN: It seems to me that this is simply a variation of a topic we discussed at length this morning; namely, how are you going to interest people whose primary concern is not computing? We are talking again about people who will not primarily be users but who simply want an appreciation of the subject.

ANDREE: A little while ago I asked for some specific help on a specific problem and I got quite a bit of it. Let's see if we couldn't do that again. Let me pose a specific problem. The correspondence section at our university has asked me several times whether I could create a correspondence course in computing. My answer up to now has been that I don't have the time, but suppose someone has the time, is competent, and wants to design a correspondence course for adults. What would you put in such a course?

GRUENBERGER: We even have precedents for that. Computer Usage has such a course that is offered commercially, and USAFI has a correspondence course. So someone has

designed the content that you are talking about, Dick. Why should such a course be much different from the one you would give if you had the people under your thumb?

ATCHISON: There is clearly a need for correspondence courses. We continually received such requests at Georgia and now at the University of Maryland.

SHADER: If we could facilitate such education, we would be accomplishing a great deal. That brings us back to the question: what should be the content of such a course? I don't think we have to be too specific; the main problem is one of attitude and philosophy.

MILLS: The course that Computer Usage offers is the sort of course that one would take if he were trying to break into the field. I don't think we're talking about a course like that.

We have adult education in most of our high schools in this city, and people can request almost any kind of course. There was such a course at Webster High School, and they had a great deal of trouble getting someone to teach it. Now they're having trouble getting people to attend it. Possibly their main problem is inadequate publicity. I'm simply pointing out that if there is a great demand for knowledge in computing, this is one avenue that can supply it.

FINERMAN: There seem to be only two avenues for reaching these adults. One is through the adult education classes, and the other is by correspondence. In either case I would think it would revolve around the topic of appreciation.

SHADER: You can make some general statement about such courses. They have to be on a fairly popular level, and they can't require much homework. The people who enroll in such courses are usually tired in the evening, and you can

perhaps hold their attention for three hours at the extreme limit, although two seems to be better.

FINERMAN: And you have to concentrate on what is essential.

MILLS: So we should be outlining what you want to teach them.

What are some of the other things that need to be done along these same lines? We've had discussions of this in SHARE. We have many computer meetings going on almost continuously: SHARE, GUIDE, the joint computer conferences, the ACM conferences and others. Why could there not be at these meetings a room devoted to lectures and/or demonstrations on computing open to the public and advertised as such? I would think that the public might be quite eager to attend such shows if they were publicized along with the regular publicity that goes with every conference.

I think this was done successfully at one SHARE meeting.

GRUENBERGER: That's a tremendous idea. The idea has come to me many times but from the other way around. We have about four large conferences a week going on in Los Angeles; we might have, for example, all the psychiatrists in the world congregating here for a week-long meeting, and the thought always passes through my mind as I'm reading about it in the paper: wouldn't it be nice to go down there and hear a little about their discipline? And no one ever does it.

GALLER: Wouldn't you worry that maybe too many people would show up?

MILLS: That would be the greatest thing that ever happened.

WEIZENBAUM: I would worry more that you would get the lunatic fringe showing up at such things. I think educational television is the optimum means of reaching the public.

MILLS: I don't understand what you mean about the lunatic fringe. The nuts wouldn't come out to sit through meetings. What would they be a lunatic about? Maybe that's the way to ask it.

WEIZENBAUM: I don't know, but I would fear that you would get many people who would get excited about whatever it is people get excited about these days. Sometimes it's bomb shelters; sometimes it's "the computers are taking over."

MILLS: Maybe we need them there. Maybe they really need help.

WEIZENBAUM: I say you should reach them with educational television.

MILLS: Those are the people who do not watch educational television. I'm interested in everyone who should learn about computers; I would run the risk that a few nuts might show up, and we would have to take care of them.

WEIZENBAUM: I'm talking about people who have time to spend, perhaps twenty hours or so. There are probably many such people who would like to know about computers. We cannot handle them in any case, but we certainly can't handle them economically. There would have to be lectures in large auditoriums given in 40 major cities, and there aren't enough computer people who have that kind of time. We could, however, seek out two or three outstanding teachers; they could work very hard to prepare an outstanding series of lectures which could be put on film for television. That would be economical. It would also be the best way to do it. Such a message could reach a number of people, and many of them are important.

EVANS: You would also reach a different audience. The sort of people who would come to a public hall at an AFIPS conference are not the same people who might be reached through television.

GRUENBERGER: I'm certainly in favor of that, but Roger's suggestion still has merit. We can take care of the lunatics who might show up by assigning someone to take them off in a corner and keep them happy. But the people who would like to hear something for an hour and learn something--anything--about computers should be catered to. They are a different group from the ones you would meet on television.

WEIZENBAUM: I still like the idea of educational television. Notice that it doesn't have to be on our ETV channels; it could be on any station. What intrigues me is that you might have 20 lectures done superbly and make an impression.

WHITE: It would certainly have to be done well because you are competing with Petticoat Junction.

WEIZENBAUM: Mosteller did that series on statistics. It was educational television at its best. It has become something of a legend. The films are shown over and over again, and they are good. He is a master statistician and a master teacher.

WHITE: There's a difference between education and propaganda, and part of what we want to do is propaganda.

DAVIDSON: There would be merit, I think, in continuing to differentiate between a message sent to the general public and one sent to selected groups within the general public. We have talked about science writers, labor leaders, and teachers. It would seem to me to be more profitable to tailor the approach to one of these groups than to have a shotgun approach to anyone in the public who wants to come.

MILLS: I thought you were going to approach this the other way around. Fred mentioned that there are meetings almost daily in a large city like this of selected groups of people like the psychiatrists. Would it be possible

to arrange to present a lecture or a tutorial on computing to these groups at their meetings?

GRUENBERGER: That is the function of the program chairman of the other group who should be looking into that anyway.

MILLS: But he obviously isn't doing it, and we should offer this service. If we were to address an offer to program chairmen of conferences in other disciplines, I doubt that they would turn it down.

ANDREE: By the time you know of a conference in any other discipline, its program has been set for months.

GRUENBERGER: That's all right; you make them the offer, and they can include it in their next conference.

FINERMAN: The last convention of the psychiatrists did have a talk on computing.

WEIZENBAUM: Yes, in fact, I have addressed conventions of psychiatrists twice now. Each time I addressed the American Psychiatric Association I had a rather sick feeling.

GRUENBERGER: I imagine they wanted to hear about your psychiatric program, didn't they?

WEIZENBAUM: No. What gave me a bad feeling was that I felt they didn't know enough. You can imagine this the other way around, if a surgeon addressed a group of computer people for an hour to tell what he does. The education must go on at a much more general and broader level.

I keep coming back to the idea of educational television. Films could be shown to small groups by projector or could be shown over television. Such films could even be geared to a specific computing system so that groups could rent the films for a period of time and receive with them instructions on how to prepare cards for that particular computer. This idea is not too improbable even for television, since I think eventually the audience will be invited to participate--for example, by means of punched cards.

ATCHISON: Would the development of a series of films like this be a worthy project for the AFIPS education committee?

WEIZENBAUM: That would surely be worthwhile. The fundamental problem is energy, not money. When you stop to think of it, the fundamental problem with every controversy we've had in this room today has been people. It always comes down to finding the right people who have the energy and the time.

GALLER: I agree that the basic problem here is not money. The U.S. Office of Education could probably be approached to finance such a project.

WEIZENBAUM: That's why you need a project that has built-in amplification and leverage. Films and television have the amplification factor, whereas public lectures do not. That's why a man like Mosteller gives up many, many hours of his valuable time. He has to know that his efforts will be multiplied over and over again.

SHADER: Would you give me the names of some senior people in the field whom you think would be acceptable for a project like this? They would have to be knowledgeable and acceptable as performers in front of the camera.

WEIZENBAUM: Let's start with Allen Newell.

SANDERS: Marvin Minsky.

WEIZENBAUM: McCarthy--he comes across pretty well on television.

GRUENBERGER: If I'm not mistaken, you've been in front of the cameras, haven't you, Joe?

GALLER: Are you naming people who would be expected to make up the content of the film or just present it?

DAVIDSON: Yes! They have to be able to do both sides of the thing.

WEIZENBAUM: You can't just give these people a script; that won't work.

SHADER: I assume we can get people to do all the rest of the work. We can at least assume that that problem will be solved. We are talking about the people who must appear before the cameras.

WEIZENBAUM: I wouldn't be surprised if Charlie Davidson would be a candidate for this job as well.

WHITE: I would suggest that the important thing is not to worry about the man's name in the field as long as he is good at it.

GRUENBERGER: Let's add Cliff Shaw's name to the list. He happens to be magnificent in front of a camera. He is not only superb to work with on a set, but he does have stature in the field. Another one is Dick Hamming.

DAVIDSON: I'd like to suggest Arthur Kahn from Westinghouse who gave a talk at ACM in Washington about the type of course in computer appreciation he is giving in-house for Westinghouse. He has a collection of cartoon slides, numbering about 60, which is tremendous.

GRUENBERGER: Let's also add Andy Kinslow, who used to be with IBM.

WEIZENBAUM: But a good one who is with IBM is Charlie DeCarlo. Let's also add Nat Rochester.

WHITE: And Jim Babcock.

WEIZENBAUM: So there you are--you ask for some names, and you get quite a list.

CANNON: Not all these people would have the time.

WEIZENBAUM: But that's just the point. Nobody had less time than Mosteller!

WHITE: That's right. Almost anyone would agree to take on a project like this.

I'd raise again the question of whether the people for such a project have to be well-known as long as they are good? Though they might be well-known within our profession, they won't be known to the audience anyway, so the emphasis should be on good presentation.

SHADER: That's true, but the man who is well-known in the field--as Mosteller was in his--carries not only the weight of authority but acceptance at all levels.

GRUENBERGER: Senior stature solves many problems for you because you are getting it from the horse's mouth. Somehow that air of authenticity cannot be conveyed by someone else unless he is a superb actor, and that we don't need.

SHADER: There are certain problems in putting together a series of films, whether it be done by AFIPS, a foundation, the Government, or whomever.

GRUENBERGER: Do you want still more names for that list? If so, I would suggest you don't overlook Mr. Krehbiel.

WEIZENBAUM: When you come right down to it, Mosteller was not well-known at all to the students who saw those particular programs; but he became well-known within his field because he is so good.

SHADER: For that particular series of films there was no problem in obtaining the money.

WHITE: You can fund such a project on the basis of a certain number of big names (who are known to be good), and, then, you can fill in the gaps with others.

CANNON: I would expect that if the quality were as high as you seem to be talking about, a series of films could serve more than one group. We have been talking in terms of adult education, but the same films would be highly useful in the training of teachers and others.

(The meeting adjourned.)