DOCUMENT RESUME

ED 290 457 IR 013 113

AUTHOR	Mioduser, David; And Others
TITLE	Teaching Programming Literacy to Non Programmers: The Use of a Computerized Simulation. Technical Report No. 15.
INSTITUTION	Tel-Aviv Univ. (Israel). Computers in Education Research Lab.
SPONS AGENCY	Israel Ministry of Education, Jerusalem.
PUB DATE	Feb 85
NOTE	27p.; For related reports, see IR 013 108-115.
PUB TYPE	Reports - Research/Technical (143)
EDRS PRICE	MF01/PC02 Plus Postage.
DESCRIPTORS	Computer Literacy; Computers; *Computer Simulation; Computer Software; Concept Formation; Foreign Countries; *Information Processing; *Instructional Effectiveness; Instructional Material Evaluation; Intermediate Grades; Models; Preadolescents; Pretests Posttests; *Programing
IDENTIFIERS	*Israel

ABSTRACT

The Transparent Computer, a computer simulation designed to teach programming concepts to non-programmers, provides a graphic representation of the computer and illustrates information flow within the units of the computer. The emphasis of the simulation is on the acquisition and application of concepts and comprehension of the structuring of output through the execution of the program's statements, not on learning the technical and syntactic aspects of a large computer language. A study was conducted with fourth and sixth grade students to determine whether this type of simulation could be used to teach basic programming concepts to the non-programmers population. All students were found to be able to program the Transparent Computer and run their own programs within a short period of time, whether or not their pretests had shown any previous programming experience. The study results suggest that the Transparent Computer is the kind of concrete model whose transparency, interactivity, concreteness, and simplicity provide an efficient method for teaching basic programming. A 13-item bibliography is provided. (EW)

THE COMPUTERS IN EDUCATION RESEARCH LAB. המעבדה לחקר יישומי מחשבים בחינוך

ED290457

TEACHING PROGRAMMING LITERACY TO NON PROGRAMMERS: THE USE OF A COMPUTERIZED SIMULATION

David Mioduser, Rafi Nachmias, David Chen

Technical Report No. 15

February 1985

IR013113

TEL AVIV UNIVERSITY SCHOOL OF EDUCATION בית הספר לחינוך אוניברסיטת תל אביב

UNIT FOR COMMUNICATION & COMPUTER RESEARCH IN EDUCATION היחידה לחקר תקשוב בחינוך

THE COMPUTERS IN EDUCATION RESEARCH LAB המעבדה לחקר יישומי מחשב בחינוך

TEACHING PROGRAMMING LITERACY TO NON PROGRAMMERS: THE USE OF A COMPUTERIZED SIMULATION

David Mioduser, Rafi Nachmias, David Chen

Technical Report No. 15

February 1985

This paper has been prepared in the Computers in Education Research Lab., the Unit for Curriculum R&D, School of Education, Tel Aviv University, with assistance from the Israel Ministry of Education. The authors wish to thank the team members that helped at different stages of this work.

School of Education Tel Aviv University Tel Aviv 69978 Israel

"If I ordered a general to fly from one flower to another like a butterfly, or to write a tragic drama, or to change himself into a sea bird, and if the general did not carry out the order that he had received, which one of us would be in the wrong?" the king demanded "The general, or myself?"
"You," said the prince firmly.
"Exactly. One must require from each one the duty which each one can perform"..."I have the right to require obedience because my orders are reasonable."
(Antoine de Saint-Exupery , "The Little Prince")

1. INTRODUCTION

In 1956, only ten years after it has been constructed, the first digital electronic computer was brought into the Smithsonian Institute in Washington, as a museum piece. No more than thirty years have passed, but the circle of computer users has widened to many millions. The computer technology still continues to develop rapidly and one cannot surmise what the future form of communication with the computer might be, or the level of usage of programming languages as we know them today. However, it appears that basic concepts regarding communication with the computer and the process starting with issuing a command and resulting with an output, will be among the milestones of culture in the computer-based society of the near future. The lack of a language common to the entire population may increase even more the existing gap between those intimidated by the new technology, and those who master it.
Several groups of computer users, at different levels, may be distinguished: A small percentage of the population who employ programming and software development professionally (programmers); others, who are amateur programmers in their free time; and the majority of the population, not dealing with computer programming professionally, but nevertheless have to function in a

computer-based society (non-programmers). The uncertainty regarding the nature of the future interaction with the computer by the population at large is reflected by the lack of a clear definition of the contents which must be learned by these individuals, as a part of their basic technological education. A typical example is the "Basic Course in Programming", which, in many cases, has been suggested as the hasty answer supplied by those refusing to "lag behind", for preparing their students for life in the computerized environment (Kurland, Mawby & Cahir, 1984).

A number of problems arise during the teaching of programming to beginners:

- Learning a programming language is a time-consuming activity. Learning involves the acquisition of a large number of skills, concepts and facts in various domains. The chances of achieving a significant level of mastery of a programming language within the allocated time of the average programming course thus seem rather slim (Nachmias, Mioduser and Chen, 1985).
- The contents of the conventional programming course are focused around the syntactic and semantic aspects of the programming language (which is most often LOGO or BASIC), at a basic level. At the completion of the course the student has mastered a limited number of statements and is capable of writing simple programs. However, the teaching does not normally include concepts and skills such as the process of problem solving with programming language (understanding the problem, defining the alternative solutions, writing the program and debugging it), the ability to construct a mental model regarding the process taking place while the program is executed ("Runnable mental model", Collins & Gentner, 1981), and the components of "the programmer's thinking", beyond the syntax of a given language (the "pragmatics", described by Pea & Kurland, 1984).

Most often, the student must develop these skills on his own.

- The computer technology consists mostly of a whole of tiny components, whose external features do not reflect in any way the nature of the process which they carry out. Children are capable of exerting much imagination when considering and discussing the "power" or "motive" behind the computer (Turkle, 1984). Often, however, they have no clear ideas regarding the functional structure of the system (the various units and their functions), nor of the processing and information flow within it. Such knowledge has been reported lacking in a considerable proportion of programming students. It would seem that the exposure of students to computer programming is not, in itself, sufficient for preventing misconceptions regarding its mode of operation, or correcting them (Anderson & Klassen, 1981; Mawby, Clemnt Pea & Hawkins, 1984). Yet, the acquaintance with the machine which is being programmed and how it actually works, appears to be desirable knowledge with which the intelligent computer user should be provided.

The present study therefore undertook to examine a proposed set of concepts, skills and knowledge related to the basic principles of computer programming irrespective of a specific programming language wich we suggest to call Programming Literacy. This may become a basic discipline with which the entire population should become acquainted.

Such a discipline should incorporate two basic content domains:

A. Knowledge of the computer and its mode of operation: Comprising of acquaintance with the various functional units of the computer, the flow of information among them and the processes which turn the program and data fed into the computer into the desirable result.

B. Comprehension of the concept of computer program: This includes perceiving the computer as an instrument operated through a series of commands received from a program which has been written by man, in order to solve a specific problem; understanding that a program is a series of instructions which are kept in the computer's memory, in an appropriate language; understanding the process through which a program is being executed, following the sequence of individual commands. All the above should be independent of syntax and semantic aspects of a specific programming language.

Teaching the population at large the above contents, as a basic discipline, may contribute to the de-mystification of the computer and discarding of prejudice regarding its mode of operation, thus serving to turn the computer into a natural component of the man-made contemporary environment.

We propose to employ a computerized simulation of information flow between the various units of the computer, as an auxiliary tool in teaching Programming Literacy to the entire population. This simulation, to be described below, has been named "The Transparent Computer". The present report describes a study in which this simulation has been implemented.

The goals of the study were:

1. Examination of elementary school students' ability to write and run programs, using the computerized simulation.

2. Comparing the degree of understanding of Programming Literacy concepts by children, before the operation of the simulation, and following it.

2. METHOD

2.1 The Transparent Computer: A computerized simulation

"The Transparent Computer" is a computerized simulation of the information flow between the various units of the computer, intended for teaching the Programming Literacy concepts to non-programmers. The simulation and the textbook acompanying it have been developed at the Computer in Education Research Lab., Tel Aviv University (Mioduser, Nachmias, Blau & Chen, 1984). The detailed description of the simulation and the rationale for its development appear elsewhere (Mioduser, Nachmias & Chen, 1984). The following section presents a general description of the simulation:

A. Graphic representation of the simulation: When operating the simulation, a graphic representation of the system appears on the computer's screen (Fig. 1):

The input unit - through which commands are issued and data fed into the computer. The working window and statements list appear in the frame.

The memory unit - In which statements and data are stored, within 20 memory cells.

The processing unit - To which statements are sequentially called for execution,and computations are performed.

The output unit - This frame substitutes the INPUT frame, when statements are executed. It actually represents the computer's screen, on which the results of the program appear. The frame consists of two windows: A graphic window, for drawings, and a text window, for numbers.

Working state: The bottom line on the screen lists the various working states of the simulator. Figure 2 presents the changing windows at the bottom right-hand corner of the screen, while each option is being operated:

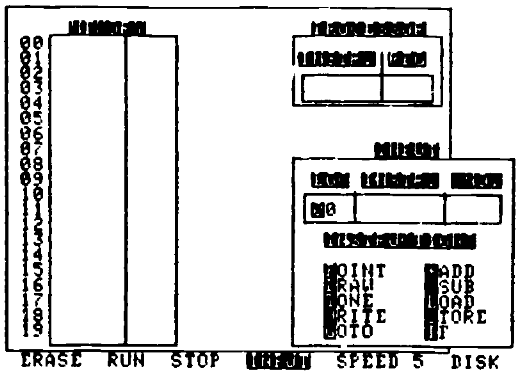

Figure 1:
Graphic representation of the different computer units.

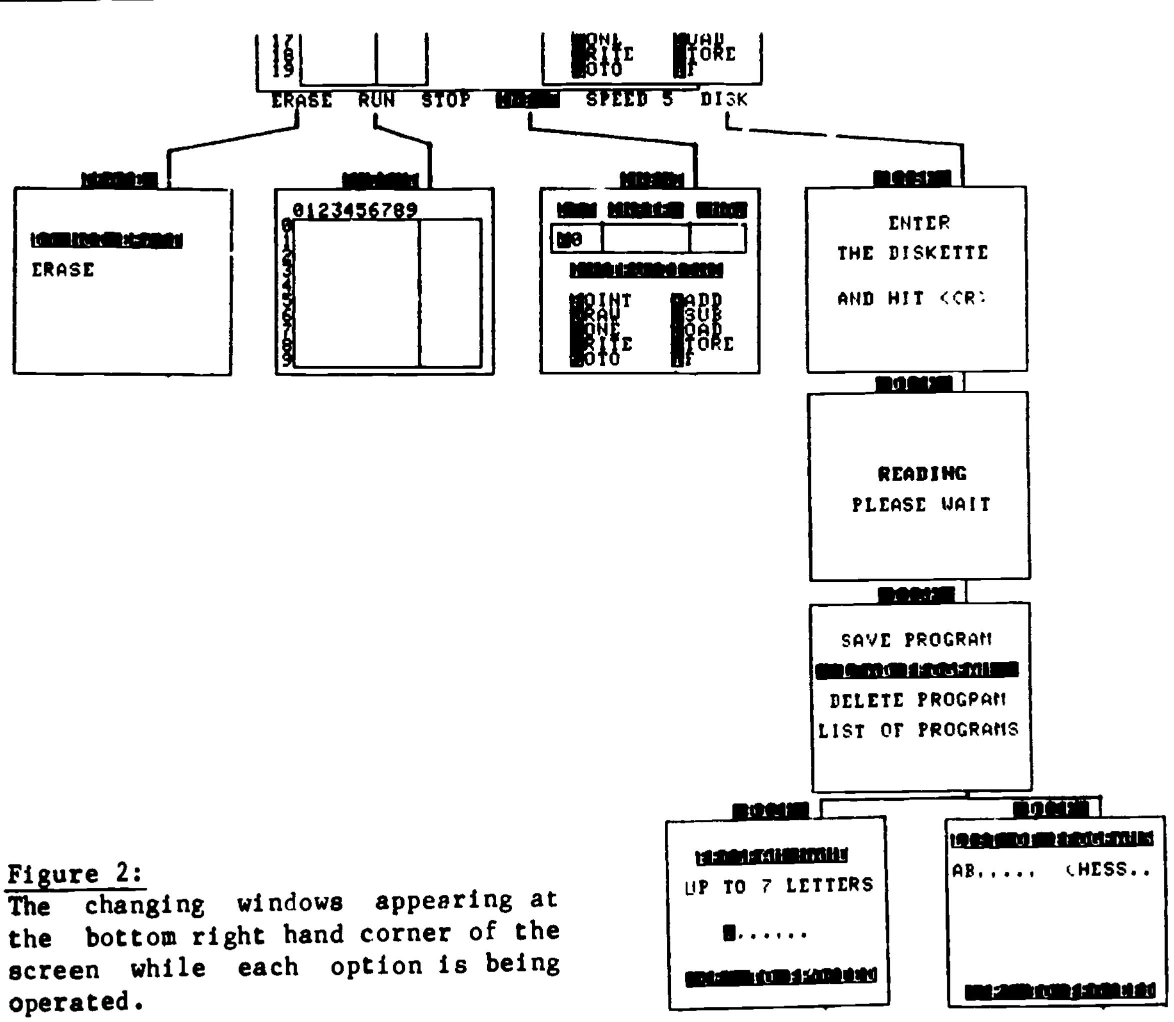

Figure 2:
The changing windows appearing at the bottom right hand corner of the screen while each option is being operated.

"INPUT": for entering statements and data; "RUN": for running a program; "STOP": for stopping the execution of a program; "SPEED": for determining the speed of the simulator operations; "DISK": for interaction with a disk, including four options - saving, loading, deleting and listing the programs; "DELETE": for deleting the program which is currently in the memory.

B. Information flow between the units: This is illustrated when operating "The Transparent Computer." When storing statements in the memory and running the program an animated process appears on the screen showing statements and numbers moving between the units involved in that operation. It is also possible to freeze the activity of the machine at any moment. When running the program, the "control arrow" points to the operation which is being executed at that moment, thus illustrating the flow of the program.

C. The levels of complexity while working with the simulation

The student's work with "The Transparent Computer" is carried out at a number of levels:

Level 1: Writing and running basic programs:

Through operating the software, accompanied by the working notebook, the student learns about the various units of the computer, how statements are being written down, and how information flows from unit to unit. At this level, the student uses action commands (Luehrmann, 1983), which include printing, graphic, sound and computation commands.

Level 2: Variable and loop:

Here the student learns to use variables as parameters of action commands. The visual qualities of the simulation allow a vivid representation of the usage of the variable in a program, including both the process of changing the value and of its effect on the result in the output. The jump statement (GOTO) serves for generating infinite loops. The combination of the variable and the infinite loop enables the production of more complex and rich output, using simple and short programs. At this stage the student is expected to show a deeper understanding of the meaning of the statemets, and the logic aspects of the program structure.

Level 3: Conditioning:

At this level the conditioning statement is added, permitting to stop an infinite loop. and branching the program. "The Transparent Computer" makes it possible to follow the movement of the control arrow from one part of the program to another, according to the specified conditions.

Level 4: Transfer of information from accumulator to memory:

The statements STORE and LOAD are added at this stage. These are inteded to store in the memory the value which is currently in the accumulator, and the reverse operation of loading a value which is stored in a memory cell, into the accumulator. Here the meaning of the variable is expressed at a more complex level; the memory cell is defined as a variable, in which values are stored and from which they are loaded, according to the demands of the program. The contents of these cells serve as parameters in the program statements, and may be manipulated through the conditioning statement. The present study focused on examination of the students' performance at the first two levels.

2.2 Subjects:

Four elementary school classes (2 4th grade and 2 6th grade), in a high socio-economical residential area in Tel Aviv participated in the study . There were 43 4th grade students and 43 6th grade students. Of the total 86 students, 64 had no prior knowledge of programming (36 4th graders and 28 6th graders). The other 22 students (7 4th graders and 15 6th graders) had some basic knowledge in programming, acquired either through participation in a programming course, or while working on a personal computer at home.

2.3 Data Collecting Instruments:

A pre/post test served for data collection. It comprised of 33 multiple-choice questions, examining levels of knowledge, comprehension and application of the Programming Literacy concepts. The questions were divided into the following 6 subjects, according to their contents:

1. Basic operations of information processing (4 questions): Students were requested to relate each of the information processing operations (input, storage in memory, processing and output) to one sentence, describing the operation. For example, the student was asked to determine to which of the operations "presenting a drawing or text on the screen" related.
2. Information flow between the various computer units (5 questions): In each question, two units were presented, and the flow of information between them; the student had to indicate whether the described information flow is correct, or not. For exmple, the student was asked to comment on the correctness of the following statement: "when the computer works, information flows from the output unit to the input unit'.

3. Functioning of the various computer units (7 questions): Students had to indicate whether the functioning ascribed to each unit is correct. For example, in the sentence "The memory unit serve to execute forgotten operations".

4. The essentiality of instructions issued by man for the computer's operation (5 questio ,tudents had to indicate whether statements such as "The computer requires instructions only when it has to print instructions on the screen", were true or false.

5. The concept of "computer program" (4 questions): Students were asked to indicate whether the reference to this concept is correct, in statements such as "In a good computer program, the sequence of the statements is not important".

6. Communication with the computer through a language which has been adapted to the machine (8 questions): The students were given a description of an imaginary computer, which "understands" a language comprised of only three statements "+"; "/"; "*". They were asked to examine several "programs", consisting of the above and/or other symbols (e.g. "5*5+3", or "///**///") and indicate which of these program may be run without errors in the imaginary computer.

2.4 Procedure:

All subjects were given the pre-test. One 4th grade and one 6th grade were assigned to the experimental group (total of 44 students), and were given 5 double-lessons of computerized simulation (approximately 10 hours). The other two classes (total of 42 students) were given no lessons between the two tests, and served as a control group. The simulation was conducted with Apple IIe microcomputers; 2-3 students shared one computer.

The first lesson consisted of general introduction, including a description of the various units of the computer. The students then continued with independent work, guided by the special working notebook. Following the completion of learning, all students were administered the post-test, identical to the pre-test.

The independent variables in the study were:

A. Learning: The experimental and the control groups.

B. Age level: Grades 4 and 6.

C. Achievements: Each student was assigned to one of three levels, based on semi-annual grades in arithmetics and Hebrew; students were thus divided into those with high, intermediate or poor academic achievements.

D Previous knowledge in programming: The two categories were with and without previous knowledge.

E. Subject´s sex.

3. RESULTS

3.1 The pre-test:

Table 1 summarizes the results of the pre-test. The averag. scores in each of the 6 subcategories of the test, as well as the total score are presented for the entire sample, and for subgroups divided according to achievement levels and according to previous knowledge of programming.

TABLE 1: MEAN SCORES IN THE PRE-TEST (MAXIMAL SCORE = 100)

		N	processing stages	information flow	units function	computer needs instructions	computer program	language adapted to the machine	overall score
all students		86	41	36	45	62	60	26	43
grade	4	43	29	30	37	53	53	14	34
	6	43	52	42	53	71	67	39	52
academic achieve ments	1	24	19	21	26	41	36	10	24
	2	27	30	30	41	57	62	17	37
	3	39	60	48	59	75	71	40	57
progra mming	Y	64	28	23	33	54	50	15	32
	N	22	78	72	80	85	91	59	76

academic achievements: (1) poor; (2) intermediate; (3) high
experience in programming: (Y) yes; (N) no

The following findings are revealed by Table 1:

- The scores of the entire sample indicate a rather modest overall knowledge of the concepts studied by the test (around 40%). A trend was revealed according to which students whose academic achievement level was higher, also scored higher in the pre-test. Considerable differences were observed between students who had previous knowledge of programming, and those who had none.

- All students received relatively high scores in questions regarding the necessity of commands issued by man. and questions about the concept of computer program. On the other hand, questions concerning information processing, the flow of information between the different units of the computer, the functioning of each unit and the need for a language adapted for the machine, yielded poor scores. Dividing the subjects according to class, academic achievement level and previous experience with programming reveals similar differences between the scores for questions regarding instructing the computer and the concept of computer program, and the scores for the other questions.

- A considerable difference between students with and without previous experience in programming was noted with regard to the concept of "a computer program". Whereas children without any previous knowledge demonstrated an average level of 50%, students with previous experience in programming scored around 90%.

- Examining the answers to single items revealed relatively high internal consistency within certain groups of items. Those items related to the memory unit and its function in the execution of the program were answered correctly by a high percent of the students (approxiamtely 66%). Conversely, only about one third of the students replied correctly when asked about the processing unit and its role in the information processing process. The items regarding the necessity of giving instructions to the computer was answered correctly by a considerable proportion of the students (60-80%, depending on the question), with the exception of one item, to which only about 35% of the students gave the correct answer (students had to say whether the following statement was true or false: "An instruction to the computer must be given only if one wishes to see the instruction on the screen").

3.2 Differences between the pre-test and the post-test:

Table 2 summarizes the mean scores of the experimental and the control groups, with regard to each of the 6 sub-categories of questions, as well as the overall test score.

TABLE 2: MEAN SCORES OF THE EXPERIMENTAL ANF CONTROL GROUPS, IN THE PRE- AND POST-TESTS

Group	N	processing stages Pre	processing stages post	information flow Pre	information flow Post	units function Pre	units function Post	computer needs instructions Pre	computer needs instructions Post	computer program Pre	computer program Post	language adapted to the machine Pre	language adapted to the machine Post	overall score Pre	overall score Post
experimental	44	45	79	39	75	49	73	61	83	59	76	26	53	44	71
control	42	36	57	33	44	41	56	63	80	62	74	26	43	42	57
		--	**	--	**	--	**	--	--	--	--	--	--	--	**

(**) significant differences between the scores of the two groups at $P<0.01$

Whereas no differences between the groups was found in the pre-test scores, the differences in the post-test scores are high for all sub-categories, as well as for the overall test score.

Large differences between the pre- and post-test scores are noted for items related to stages of processing, the units of the computer and their functioning, and the flow of information during the execution of the program. On the other hand, the differences between the groups for items reflecting general knowlege about giving commands to the computer and the concept of computer program, are much smaller.

Low scores were seen in items concerning the need for a language adapted for the machine, in both tests and both experimental and control groups.

3.3 The sources of variability:

The results of the analysis of variance, examining the effect of grade, learning with the simulation, academic achievements, previous experience in programming, and the subject's sex on the students scores in the pre- and post-tests, are presented in Table 3.

TABLE 3: THE SOURCES OF VARIABILITY IN PRE- AND POST-TEST SCORES

	grade		academic achievements		programming experience		sex	
	pre	post	pre	post	pre	post	pre	post
processing stages	**	**	**	**	**			
information flow					**	**		
units function	*	**		**	**	*		
computer needs instructions	**		*		**	**		
computer program				**	**			
language adapted to the machine	**			**	**			
overall score	**	**	*	**	**	**		

* $p < 0.05$
** $p < 0.01$

The major variables affecting the students' scores in both pre and post-tests, are previous experience in programming and academic achievements. The grade variable has an effect mostly on items related to stages of processing, the functioning of the various units and the flow of information from one to another, and the overall score. The subject's sex has no effect whatsoever on the scores.

3.4 Preliminary impressions concerning the operating of the simulation

Observing the students in their classes showed that most students, in both age levels and all academic levels, succeeded in operating the simulation, writing programs producing various types of output (graphics, sound and numeric), and running those programs. Students worked mostly at the first two levels (running simple programs using action commands, and using variables and infinite loops). Both 4th and 6th grade students managed to write and run a program within the first lesson. Most of the programs written during the study included drawing or sound commands, adding and subtracting, and the statement for generating an infinite loop. Students wrote two kinds of programs with loops: The first was a fixed program, repeating itself until stopped. Students often wrote a number of statements arbitrarily, not planning in advance, and "locked" this series with the GOTO statement. The other kind was an infinite loop with variables. Here, too, programs were mostly improvised, and the students could not predict the result of changing the values during the repetition of the program. The majority of the students succeeded in writing programs of the first kind, but only a few of the 6th grade students could generate loops with variabes, plannig in advance the goal to be reached by the changing of the variables.

The language with which "The Transparent Computer" may be programmed allows the application of a number of basic concepts in programming, such as variable, loop, and conditioning. The limited duration of the experiment prevented the transition of students to the higher levels of working with the simulation, and the present study therefore did not examine the acquisition of these concepts.

4. DISCUSSION AND CONCLUSIONS

A large number of studies conducted in the recent years point to the great difficulty in teaching a complex activity such as computer programming, to the population at large. Acquisition of the skills and knowledge required for becoming a competent programmer, demands much time and effort. However, other possibilities exist, as suggested by Kurland et al.: "...We believe that there are ways to teach fundamental programming and computer science concepts in the normal classroom as a solid foundation for students' future interactions with computers, whether they continue to use computers for programming, word processing, or other information management purposes" (Kurland, Mawby and Cahir, 1984, p. 17).

In the same venue, the present study has proposed an alternative approach to teaching basic concepts of programming to the non-programmers population: elementary school students, and people who may never engage professionally in computer programming. The operation of a computerized simulation was examined experimentally in this study, and the implications of the finidings are presented below.

4.1 <u>The existing knowledge regarding basic concepts of programming:</u>

The results of the pre-test indicate that the subjects had an intermediate level of knowledge of the basic programming concepts which were tested (overall mean score, 43 out of 100). Analysing the items contents reveals relatively high scores for questions reflecting general knowledge about the interaction with computers (such as the necessity of issuing instructions to the computer, and the concept of a computer program). Poor scores, however, were seen when the items related to more specific knowledge regarding the functioning of the computer's units during the execution of a program, or the need to use a language which is adapted to the machine. These findings may be interpreted as indicative of some "computer culture", which is spreading among the population, and outside the formal education system. The contents of such literacy are centered around the more general levels of interactions with the machine: Knowing that the computer is a machine which obeys instructions; t' those instructions are given by man, and that the sequence of instructions is the program which the computer executes in order to produce the required output. The other conclusion which may be drawn from the above findings is that specific concepts are not yet part of this computer culture, and therefore need to be acquired through a systematic learning process, e.g. concepts regarding the funcioning of the computer which is being programmed, its various units and the need to use a language which is compatible with the machine.

The present findings may be interpreted in light of the definition of learning as a process of assimilation of new knowledge within an existing conceptual framework (Weltner, 1973; Mayer, 1981). Students appear to have related the new concepts to the previously existing knowledge, associations and images. For example, about two thirds of the students replied correctly to all questions regarding the computer´s memory and its role in storing programs and data. Likewise, items describing the computer as a machine needing instructions in order to carry out the required operations, were answered correctly by 60-80% of the students. It would seem that students have some images concerning "the obedient machine" and memory, which helped when new concepts were introduced. The poor score achieved when students were requested to indicate whether the statement "the computer requires instructions only when it has to print instructions on the screen" was true or false, supports the above interpretation. Most children, especially those without any previous experience in programming, were misled by the obvious aspect of inputting (typing the command - the appearance of the command on the screen), and based their judgemant of that statement on this aspect. Only one third of the studets answered the above item correctly. Questions related to units, or concepts which cannot be easily related to images or interpreted within the existing knowledge, resulted in poor scores. For example: Only about one third of the students correctly replied to questions concerning the processing unit. The above findings suggest that it would be advisable to emphasize the developing of those teaching strategies which provide a suitable conceptual framework for assimilating the basic concepts of computer programming.

4.2 Comprehension of basic concepts of programming by children studying with the simulation:

The present study proposed a teaching strategy based on the computerized simulation, "The Transparent Computer", in which the activity is based on a minimal effort in learning technical and syntactic aspects of a large computer language. Rather, it has focused on acquisition and application of the concepts, and comprehension of the structuring of output through the execution of the program's statements.

The significant differences in the mean overall post-test score between the experimental and control groups, as compared to the lack of significant differences in the pre-test scores, point to the contribution of working with the simulation to the acquisition of the concepts by the students.

Observations conducted during the operating of "The Transparent Computer", also indicate the advantages of the proposed strategy. All subjects, without any exception, succeeded in "programming" the "Transparent Computer" and running their programs, within a short time span. Individual differences were evident mostly with regard to the quality of the products (the structure of programs written, and the level of application of the learned concepts). Some basic level of knowledge and elementary skills of operating the simulation and programming it, however, was mastered by all students. Thus, the simulation seems to be suitable for teaching in heterogenous classes, in which there are students with varying levels. Wider implementation of the simulation, accompanied by a systematic follow-up, may provide a better information regarding this issue.

As mentioned earlier, students with previous experience in programming demonstrated higher scores at the pre-test, which exceeded by far the knowledge of students lacking such experience. Nevertheless, the achievements of both experienced and inexperienced students in the experimental group became similar following the work with the simulation. The majority of students reached a reasonable level of knowledge and application ability, after a relatively small number of lessons. This strongly suggests that employing a teaching strategy based on activities such as the one proposed in the present study is advantageous for achieving the goal of teaching basic concepts of programming - Programming Literacy - to the wider population. This gains further significance in view of the difficulties, the efforts and the time required for teaching conventional programming, as well as curricular and cognitive problems involved in it.

Mayer (1981) and DuBoulay, O'Shea & Monk (1981) point to the advantages of the use of concrete models as an efficient strategy for teaching programming concepts. Interactivity, simplicity, concreteness and transparency are among the recommended qualities of a computerized model for teaching. "The Transparent Computer" is an example of a computerized model applying these qualities. The interactive aspect of the simulation was evident all along the experiment. Students enthusiastically stored input in the memory, ran programs, located and corrected errors. The model's simplicity and the ease with which it is operated, contributed to the speed of mastery of operating, demonstrated by the children.

This has usually required no more than a few minutes, enabling the children to concentrate from the start on the contents to be learned. The transparency of the simulation was of major importance where comprehension of processes and concepts was concerned. The transparency of the memory unit permitted the examination of the program, planning the exchange of statements or correcting them, and referring to specific cells and their contents in order to generate loops, etc. The "crawling" of the informati[illegible] items during the execution of the program, and the ability to follow the gradual formation of the output, allowed the location of statements responsible for undesired results and understanding the relationship between each statement and the output.

The majority of the population does not require the use of a programming language for daily functioning. On the other hand, most people will have increasing interaction with computers, based on issuing commands at varying levels of complexity to the computer. Therefore, the need arises for a clear-cut definition of the contents to be taught and the strategy for teaching those contents. The present study has proposed to teach the body of concepts defined as Programming Literacy. The use of a computerized simulation, "The Transparent Computer", has been suggested as the strategy of teaching these concepts. These two suggestions were examined experimentally, and the reported results were most encouraging. Three major implications of the present study are evident: Firstly, the "computer culture" is consolidating outside the schools, and the educational system must catch up with this development and fill up the gaps, where systematic learning processes are required. Secondly, extensive devolopment of teaching strategies and instruments, such as that proposed by the present study, is needed, in order to enable the teaching of additional contents in the computer literacy discipline. Finally, a comprehensive study of understanding and application of concepts related to computer programming (e.g. variable, loop, conditioning, etc.) among students learning through these teaching tools, is recommended.

REFERENCES

Anderson, R.E.; Klassen, D.L., "A Conceptual Framework for Developing Computer Literacy Instruction", A.E.D.S. Journal , Spring 1981.

Collins, A.; Gentner, D., "Constructing Runnable Mental Models", Proceedings of the fourth Annual Conference of the Cognitive Science Society , 1982.

duBoulay, B.; O´Shea, T., Monk, J., "The Black Box Inside the Glass Box: Presenting Computing Concepts to Novices", Int.J.Man-Machine Studies 1-, 1981 , 237-249.

Kurland, M.D.; Mawby, R.; Cahir, N., "The Development of Programming Expertise in Adults and Children", in: Kurland, M.D.(ed.), Developmental Studies of Computer Programming Skills , AERA Annual Meeting, 1984.

Luehrmann, A., "Slicing Through Spaghetti Code", The Computer Teacher , April 1983.

Mayer, R.E., "The Psychology of Learning Computer Programming by Novices", Computing Surveys, 13 , 1981, 121-141.

Mawby, R.; Clement, C.A.; Pea, R.D.; Hawkins, J., "Structured Interviews on Children´s Conceptions of Computers", Bank Street College of Education, Technical Report No. 19 , 1984.

Mioduser, D.; Nachmias, R.; Chen, D., "The Use of Computerized Simulation to Introduce the Functional Structure and Operation of the Computer", The Computers in Education Research Lab., Tel Aviv University, Research Report No. 4 , 1984.

Mioduser, D.; Nachmias, R.; Blau, A.; Chen. D., The Transparent Computer , Student Handbook, The Computers in Education Research Lab., Tel-Aviv University, Israel, 1984.

Nachmias, R.; Mioduser, D.; Chen, D., "Acquisition of Basic Programming Concepts By Children", submitted for publication, 1985.

Pea R.D.; Kurland D.M., "On The Cognitive Effects of Learning Computer Programming". New Ideas Psychol. Vol.2, No.2 , 1984, 137-168.

Turkle, S., The Second Self , Simon & Schuster, N.Y., 1984.

Weltner, K., The Measurement of Verbal Information in Psychology and Education, Springer - Verlag, 1973.