DOCUMENT RESUME

ED 262 756		IR 011 834

AUTHOR	Delclos, Victor R.; And Others
TITLE	Teaching Thinking through LOGO: The Importance of Method. Technical Report Series, Report No. 84.1.2.
INSTITUTION	George Peabody Coll. for Teachers, Nashville, TN. Learning Technology Center.
SPONS AGENCY	National Inst. of Mental Health (DHHS), Bethesda, Md.
PUB DATE	84
GRANT	NIMH-MH-38235
NOTE	21p.
PUB TYPE	Reports - Evaluative/Feasibility (142)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	Academic Achievement; Comparative Analysis; *Computer Assisted Instruction; *Computer Software; Discovery Learning; Elementary Education; Grade 5; *Intermode Differences; *Problem Solving; Research Methodology; *Teaching Methods; Transfer of Training
IDENTIFIERS	Instructional Effectiveness; *LOGO Programing Language; Tutorial Mode

ABSTRACT

Research was conducted to compare the effects of three different methods of teaching LOGO to fifth graders with a broad range of academic ability. The first method, based on Papert's idea of discovery learning, involved teacher presentation of LOGO instruction and student control of pace of learning. The second method involved the use of structured tutorials, written in much the same way as other computer assisted programs. The third method involved teaching in a mediational style, where the teacher made specific and conscious attempts to frame what was learned in the LOGO lesson in a broader context, and to bridge specific principles of learning to other situations where the same type of strategy would apply. Trends which emerged from this research suggest: (1) students involved in the discovery approach learn basic LOGO commands, but do not gain control over the turtle (a form on the computer screen) to the extent that they can predict what it will do; (2) a structured approach to learning LOGO seems better than discovery for mastering programming-specific skills; and (3) a carefully structured, mediational method seems to provide good results. (JB)

TEACHING THINKING THROUGH LOGO:

THE IMPORTANCE OF METHOD[1]

Victor R. Delclos, PhD
Research Associate
Department of Psychology
Vanderbilt University
P. O. Box 9 - GPC
Nashville, TN 37203

Joan Littlefield
Research Assistant
Department of Psychology
and Human Development
George Peabody College
Vanderbilt University
P. O. Box 512 - GPC
Nashville, TN 37203

John D. Bransford, PhD
Professor of Psychology
and Education
Department of Psychology
Vanderbilt University
P. O. Box 9 - GPC
Nashville, TN 37203

1 This research is supported by Grant No. MH-38235 from the National Institute of Mental Health, John Bransford, Keith Clayton, and Jeff Franks, Principal Investigators.

Learning Technology Center Technical Report Series

Report No. 84.1.2

TEACHING THINKING THROUGH LOGO:

THE IMPORTANCE OF METHOD

ABSTRACT

While there has been a good deal of excitement about the use of the programming language called LOGO as a means of teaching general thinking skills, recent research suggests that it has not succeeded. One reason for this may be the failure of researchers to focus on the method of teaching. Preliminary observations from a project that is examining the effects of three methods of teaching Logo (discovery, structured, and mediational) is discussed. The working hypothesis presented is that teaching Logo in a "mediational" format based on current cognitive literature will produce generalizable cognitive development in children.

TEACHING THINKING THROUGH LOGO:
THE IMPORTANCE OF METHOD

Personal computers are becoming available to more people each year. At the same time, their capacity to do complex tasks is increasing. Both at home and in the classroom, users have only begun to explore the potential applications of these powerful machines.

THREE USES OF COMPUTERS

Taylor (1980) has outlined three current educational applications of microcomputers. First, a computer can be used as a tutor. Almost everyone has seen examples of "computer assisted instruction" where a particular subject is presented step-by-step to a student. For example, the student is given a multiplication problem to solve and the program offers appropriate feedback when the child answers. The program provides drills when the student encounters a problem and does not present new material until the student masters each successive step in the program. In this first application, the computer serves as an individualized teaching machine.

Second, a computer can be used as a tool to accomplish some task more efficiently. For example, word processing (computerized typing) offers a number of advantages over conventional typing. For many people, it completely eliminates the need for a handwritten draft. Typing is easier since, on most systems, the user never has to hit the return key. To correct an error, a typist can simply hit a "delete" key instead of using an eraser or correction fluid. Also, a writer can add or delete words without retyping anything.

Third, a computer can be used as a tutee. Here students learn to program the computer to accomplish a task. The teacher places emphasis on the activity of the student rather than on the task to be accomplished. This application was the inspiration for the developers of the programming language called Logo (Papert, 1980).

Logo provides an environment where students learn a small set of simple commands such as FORWARD, BACKWARD, RIGHT, and LEFT. They use these commands to instruct a small triangular form on the computer screen to draw. The triangle is called a "turtle" and this aspect of Logo is known as "turtle graphics."

SIMPLE TURTLE GRAPHICS

As an example of basic Logo programming, consider the task of teaching the turtle to draw a square (see Figure 1). The most primitive way to do this is to decide on the length of the sides of the square (let us use 100 "turtle steps") and to instruct the turtle to draw step-by-step: FORWARD 100 RIGHT 90 FORWARD 100 RIGHT 90 FORWARD 100 RIGHT 90 FORWARD 100 RIGHT 90.

INSERT FIGURE 1 ABOUT HERE.

An easier way to accomplish the same task is to use the command REPEAT. In this case, REPEAT 4 [FD 100 RT 90] will accomplish the same task as the first set of commands. An even more efficient way to draw a square is to create a new command (called a procedure) named SQUARE, e.g., TO SQUARE, REPEAT 4 [FD 100 RT 90], END. Now, by simply typing the word SQUARE, the student can instruct the turtle to draw a square of a given size. Finally, the most sophisticated approach involves defining a square of variable size by instructing the turtle to accept an input along with the word SQUARE: TO SQUARE :SIDE, REPEAT 4 [FD :SIDE RT 90], END. Now the command SQUARE 10 will produce a very small square, SQUARE 100 a large square, and so on. This variable square command can then be used to combine squares of different sizes to form complex figures such as those in Figure 2.

INSERT FIGURE 2 ABOUT HERE.

The Logo Turtle Graphics environment is a rich and challenging one for anyone who is new to the world of the microcomputer. Everyone from preschooler to college professor, from academically less successful to intellectually gifted can enjoy and learn from time spent working on Turtle Geometry projects. Logo can be a simple introduction to the somewhat mystifying world of computer programming. Many have claimed that it is much more.

WHAT DOES TURTLE GRAPHICS TEACH?

Papert (1980) has proposed that the major benefits of Logo are that: (a) it provides powerful experiential learning (in the Piagetian sense) and (b) it enhances learning and problem solving ability. Much of the excitement about the language is due to the expectation that it can develop children's thought processes and problem solving skills. In the programming examples above, the student gets practice breaking problems into component parts, planning the design of new figures, being precise in communicating directions to the turtle, and finding out what parts of a proposed solution might be wrong. All of these approaches to problem solving are applicable to problems of many kinds.

Lately, however, some controversy has arisen over Papert's claims. Research has failed to support the hypothesis that learning Logo promotes the development of more general problem-solving skills (Euchner, 1983). In a report that summarizes work conducted by a team of researchers at Bank Street College, Pea (1983) has described three studies of 8 - 12 year old children that were designed to assess: (a) the degree of programming expertise they developed during a one-year exposure to Logo, (b) the depth of understanding of certain programming concepts (e.g., recursion), and (c) the development of planning skills and the spontaneous transfer of those skills to other problems and situations. In all three of these studies Logo was taught in a discovery-oriented environment where children were allowed to choose what they would program. The results suggest that the children did not progress very far in any of the major areas studied: programming expertise, depth of understanding of programming concepts, and planning/problem-solving skills. On a test of Logo language mastery, the mean score was only 34 per cent. Only three of fifty children scored above 75. When asked to write a program to draw a box, many of the children could not complete the task except in the most primitive way (see Figure 1).

In evaluating the depth of understanding of some programming concepts, Pea and Kurland (1983) looked at six of the best programmers in their sample. They found that some of these children did not understand some concepts even though they had used them in their own programs. They also found that these children believed that the meanings of commands could be ambiguous and still work in a program. Finally, Pea (1983) found no effects of one year of Logo training on children's ability to solve a problem that required them to develop a schedule of classroom chores -- a problem that seems to involve some of the same planning strategies as those used in Logo programming.

THE IMPORTANCE OF METHOD

The researchers cited above found that learning Logo in an open, discovery-oriented environment does not result in improved planning behavior. This does not necessarily mean that Logo cannot be a useful tool for teaching general problem solving skills. Recent evaluations of several thinking skills programs have emphasized the importance of the method of presenting material when the goal of a program is to promote the development of thought processes. A recurring theme in this literature is that no content, standing alone, can spontaneously produce generalizable learning (Bransford, Stein, Arbitman-Smith and Vye, in press; Delclos, Bransford, and Haywood, 1984; Nelson, 1983).

In current research, the authors and their colleagues have begun to compare the effects of three different methods of teaching Logo to fifth graders from a broad range of academic ability. The first method is based on Papert's idea of discovery learning. The teacher presents students with the basic elements of Logo and allows them to move at their own pace and to spend their computer time as they choose. The second method involves the use of structured tutorials, written in much the same way as other computer-assisted instructional programs. In each session the computer teaches the student several new Logo words or concepts and provides practice in their use. Children spend the rest of each lesson working on problems specifically designed to provide systemmatic practice of the new material presented that day. The third method involves teaching in a mediational style. The teacher makes specific and conscious attempts to frame what is learned in the Logo lessor. in a broader context and to bridge specific principles learned to other situtations where the same type of strategy would apply. Early results of this work suggest several trends.

Recall that one of Pea's findings was that children did not do well in tests of their programming skills following extended Logo training based on the discovery method. A phenomenon that we have noticed in many children who have been given freedom to explore the Logo environment without direction is impulsive, random play with the turtle and its capabilities -- they fall prey to what we call "turtle traps." For example, some students quickly discover that they can generate interesting patterns by turning the turtle just a little to the right or left and instructing it to move forward thousands of steps. Many children continue random variations on this theme for several classes. They never understand why the patterns look as they do and they often do not understand the effects of certain commands in their programs. Preliminary data suggest that children who follow this pattern learn basic Logo commands but do not gain control over the turtle to the extent that they can predict what it will do. These children perform adequately on an examination of command mastery, but do not do well on multiple-choice items where they must choose the drawing that a given program defines. Frequently, they will have elaborate programs in their list of new procedures. Teachers often consider complex procedures evidence of programming expertise. However, observers have noted that children often copy complex procedures from classmates but do not understand how those procedures work.

Not all children follow the pattern just described. Some very talented children have done well in the unstructured learning environment. Those who learned from this approach consistently were those who set specific structured goals and limits for themselves, even though the instructors did not. These students set out to produce a specific figure and work until they have succeeded -- a sharp contrast to the random exploration described above. This suggests that some degree of structure in the Logo learning experience is necessary to insure mastery of basic commands and proficiency in planning and writing programs. If this basic material is not mastered, transfer is a moot point: a process not learned cannot, by definition, be generalized.

Thus the present research suggests one possible reason for previous failures to demonstrate the generalizabiltiy of Logo learning: students were not gaining mastery over the language before they were tested on transfer tasks. The structured tutorials incorporated into our second teaching method provide systemmatic practice on a variety of figures and designs with each command learned. This procedure appears to increase children's understanding of Logo basics and enhance their ability to predict what a given set of commands will produce. But is mastery of the language enough to insure transfer of skills?

GOING BEYOND MASTERY

Some children have learned the basic commands and concepts of the Logo language, but few have been able to transfer these skills beyond the contexts in which they are learned. Some gifted students do not even use the skills they demonstrate in other academic areas when learning Logo. For example, in classes on Logo turtle graphics students typically write programs for procedures, give those procedures names, and then add those names to the list of programs that the turtle can execute. Anything can be used as a name for any particular procedure. One of our brightest students began, very early in the course of instruction, to generate new procedures rapidly. He named his first creation (a rectangular shape) "z", the next (an arc) "v", and the next (a hexagon) "j". Soon he began using double ("jj") and triple-letter ("zzz") names for his figures. Before long he had created some thirty or forty new procedures, but he had very poor recall of what any given procedure would draw. Whenever he wanted to use a procedure he had to check through his whole list until he came to the one that he was looking for. One day the instructor intervened, asking what he was looking for. When he said, "The one that looks like a flower," he was prompted to think of a better name for that particular figure (he had named it "yyy"). He quickly realized that "flower" would be a useful name and went on to rename all of his other procedures with mnemonic names. From then on he used descriptive names for the procedures he defined.

This child seemed to have had experience creating meaningful names to enhance memory before he came to Logo classes. Nevertheless, he did not spontaneously transfer and apply that experience to the new learning situation.

Another aspect of failure to transfer can be seen in the failure to apply programming concepts that have been mastered to new programming problems. For example, another of our gifted students clearly understood how to define procedures. He performed very well on a mastery test of Logo commands and he defined procedures when he was specifically asked to do so. However, when this child worked independently, he consistently reverted to the most primitive (and least efficient) programming style. He repeatedly gave long lists of commands to the turtle instead of defining a new procedure. As with the first child, a simple prompt to think about an easier way to get the turtle to draw a design led him to realize that defining a procedure would make his work much easier. He had failed to spontaneously transfer his knowledge of procedures to another very similar problem within the same instructional context. Thus, it is not surprising that Pea and his associates have failed to find transfer of Logo learning to problems far removed from the training context.

A MEDIATIONAL APPROACH

A structured approach to teaching Logo seems better than discovery for mastering programming-specific skills. However, even this approach does not appear to foster the development of general problem-solving skills. Children frequently do not access relevant knowledge, use available strategies, or transfer skills learned in the Logo context to other areas. The next phase of our research program will compare this structured tutorial approach with a "mediational" approach to teaching Logo.

In the mediational approach, students will be helped to learn about themselves as learners and problem solvers (Bransford and Stein, 1984; Brown, Bransford, Ferrara, and Campione, 1984; Feuerstein, Rand, Hoffman, and Miller, 1980). They will be prompted to analyze and evaluate their strategies and to generate alternatives that are more efficient. In addition, students will learn to formulate general principles applicable to their programming activities and to relate those principles to activities in other domains.

As an illustration, imagine a lesson on the REPEAT command. The teacher would review what had been taught up to that point, perhaps focusing on a specific example (e.g., a SQUARE). To generate a sense of need for a new command, the teacher might point out the amount of typing required to produce a square (see Fig. 1). Next, the repetition in the procedure would be noted and the REPEAT command could be introduced as a direct simplification of this repetitious procedure. The teacher would use other examples of familiar situations where a pattern can lead to simplification of the task, such as substituting multiplication for addition of a series of identical numbers. When students could begin generating relevant examples, the second phase of the lesson would begin. Here the children would work on programming tasks that can be done efficiently with the REPEAT command, just as they would in the structured format described above. If a child completed a given task without using REPEAT, the teacher would offer individual help, highlighting the uses and value of REPEAT.

Following the practice portion of the lesson, at least ten minutes would be devoted to a discussion. First, the kinds of problems that individual students had encountered in their work would be reviewed. Various solutions to each problem would be discussed. More examples would be introduced, and the lesson would end with a summary principle for the day, e.g., "Sometimes it is easier to complete a task if you analyse its component parts."

As daily lessons lead from class period to class period, students are taught the importance of relating past experience to new tasks. They learn to appreciate the elements of problem solving common to many different tasks. These concepts are then broadened to include problems outside the computer programming domain through bridges to other areas.

CONCLUSION

Logo is clearly an exciting, rich environment for learning computer programming and problem-solving skills, but it cannot work on its own. The method of teaching Logo needs serious attention to achieve lasting, generalizable effects. A carefully structured, mediational method seems to promise good results.

REFERENCES

Bransford, J. D., & Stein, B. S. (1984). The IDEAL Problem Solver: A Guide for Improving Thinking, Learning and Creativity. New York: W. H. Freeman & Co.

Bransford, J. D., Stein, B. S., Arbitman-Smith, R., & Vye, N. J. (in press). Three approaches to improving thinking and learning skills. In J. Segal, S. Shipman, & R. Glaser (Eds.), Thinking and Learning Skills: Vol. 1, Relating Instruction to Basic Research. Hillsdale, N. J.: Lawrence Erlbaum Associates.

Brown, A. L., Bransford, J. D., Ferrara, R. A., & Campione, J. C. (1984). Learning, remembering, and understanding. In J. H. Flavell & E. M. Markman (Eds.). Handbook of Child Psychology (4th Ed.): Vol. III, Cognitive Development (pp. 77-166). New York: Wiley.

Delclos, V. R., Bransford, J. D., & Haywood, H. C. (1984). Instrumental enrichment: A program for teaching thinking. Childhood Education, 60, 256-259.

Euchner, C. (1983, November 23). Debate grows on LOGO's effect on thinking skills. Education Week, pp. 10, 15.

BEST COPY AVAILABLE

Feuerstein, R., Rand, Y., Hoffman, M., & Miller, R. (1980). Instrumental Enrichment. Baltimore: University Park Press.

Nelson, W. L. (1983). Literature as a process: An approach for the gifted. Roeper Review, 5, 14-17.

Papert, S. (1980) Mindstorms: Children, Computers, and Powerful Ideas. New York: Basic Books.

Pea, R. D. (1983). Logo programming and problem solving. (Tech. Rep. No. 12). New York: Bank Street College of Education, Center for Children and Technology.

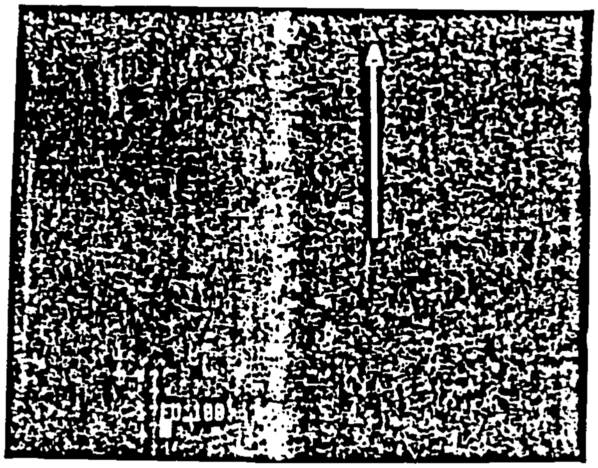
Pea, R. D. & Kurland, D. M. (1983). On the cognitive effects of learning computer programming. (Tech. Rep. No. 9). New York: Bank Street College of Education, Center for Children and Technology.

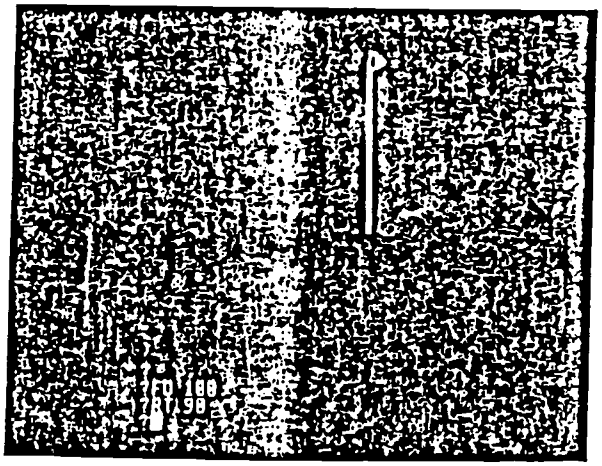
Taylor, R. (Ed.). (1980). The Computer and the School. New York: Teachers College Press.

FIGURE 1

Drawing a SQUARE with "Turtle Graphics."

BEST COPY AVAILABLE





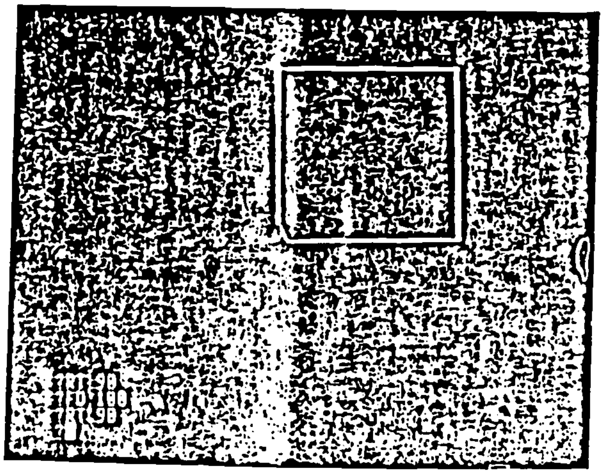


FIGURE 2

Some complex "Turtle Graphics."