DOCUMENT RESUME

ED 327 139 IR 014 758

AUTHOR	Swan, Karen; Black, John B.
TITLE	Results of Four Studies on Logo Programming, Problem Solving, and Knowledge-Based Instructional Des._;n.
PUB DATE	Mar 90
NOTE	5p.; Paper presented at the International Conference on Technology and Education (Brussels, Belgium, March 20-22, 1990).
PUB TYPE	Speeches/Conference Papers (150)
EDRS PRICE	MF01/PC01 Plus Postage.
DESCRIPTORS	Abstract Reasoning; Computer Assisted Instruction; Educational Strategies; Elementary Secondary Education; Graduate Students; Higher Education; Microcomputers; *Problem Solving; Programing Languages; *Thinking Skills
IDENTIFIERS	*LOGO Programing Language

ABSTRACT

The results of four research studies conducted with subjects ranging in age and ability from elementary to graduate school students demonstrate that Logo programming environments can be instrumental in the development of five particular problem solving strategies: (1) subgoals formation; (2) forward chaining; (3) systematic trial and error; (4) alternative representation; and (5) analogy. In fact, computing environments may be uniquely conducive to the development of such skills, in that they can support quasi-concrete, malleable representations oı tract strategies that can help learners bridge the gap between concr.,te and formal understanding. Results also indicate, however, that problem solving strategies will not be developed through Logo programming alone, but rather must be explicitly taught and practiced. (11 references) (Author/DB)

RESULTS OF FOUR STUDIES ON LOGO PROGRAMMING, PROBLEM SOLVING, AND KNOWLEDGE-BASED INSTRUCTIONAL DESIGN

by

Karen Swan

and

John B. Black

March 1990

LOGO PROGRAMMING, PROBLEM SOLVING, AND KNOWLEDGE-BASED INSTRUCTIONAL DESIGN

Karen Swan*, John B. Black°

ABSTRACT

The results of four research studies conducted with subjects ranging in age and ability from elementary to graduate school students demonstrate that Logo programming environments can be instrumental in the development of five particular problem solving strategies -- subgoals formation, forward chaining, systematic trial and error, alternative representation, and analogy. In fact, computing environments may be uniquely conducive to the development of such skills, in that they can support quasi-concrete, malleable representations of abstract strategies that can help learners bridge the gap between concrete and formal understanding. Results also indicate, however, that problem solving strategies will not be developed through Logo programming alone, rather must be explicitly taught and practiced.

INTRODUCTION

Not very long ago, computer programming was touted as a solution to the problem solving crisis in American education, a discipline through which students would automatically acquire logical thinking and problem solving skills (ref 1-2). More recently, however, such notions have gone the way of similar ideas concerning Latin and geometry. Research has indicated that problem solving abilities are not automatically acquired through computer programming (ref 3-6), and programming is accordingly being de-emphasized in computer education. Some researchers, however, maintain that higher order skills such as problem solving must be explicitly taught (ref 7). Indeed, recent investigations have indicated students' problem solving skills might be developed through programming experience if those skills are explicitly taught and practiced (ref 8-11). The research reported in this paper was designed to investigate such hypothesis. It shows that five particular problem solving strategies can be developed in students explicitly taught those strategies and given practice applying them to solve Logo programming problems. It further distinguishes between such intervention and Logo programming alone, explicit strategy training with concrete manipulatives practice, and instruction in content areas traditionally prescribed for the teaching and learning of problem solving.

STUDY ONE

Subjects. Subjects were 133 fourth through eighth grade students in a private elementary school. All subjects had at least 30 hours previous Logo programming experience.

Methodology. All subjects were given paper-and-pencil exercises testing their ability to apply six problem solving strategies -- subgoals formation, forward chaining, backward chaining, systematic trial and error, alternative representation, and analogy -- and randomly assigned by grade to one of three treatment groups receiving respectively graphics, list processing, or graphics and list processing practice problems. All subjects received training in each strategy, then asked to solve four practice problems (graphics, list processing, or both) particularly amenable to solutions involving that strategy. On completion of all six strategy units, subjects were post-tested using different but analogous exercises. Differences between mean pre- and post-test scores were examined using a four-way analysis of variance. Independent variables were test, strategy, grade level, and group. The dependent variables were scores on the strategies tests.

Results. Significant differences were found between subjects' mean pre- and post-test scores for all problem solving strategies except backward chaining, indicating the effectiveness of the explicit instruction/programming practice model for supporting the development of subgoals formation, forward chaining, systematic trial and error, alternative representation, and analogy strategies. The results also revealed

*State University of New York at Albany
°Teachers College, Columbia University

developmental differences in students' facilities for both using and developing them. Older students were better than younger ones at applying all strategies. They were also more likely to benefit from instruction in alternative representation and analogy, while younger students benefited more than older students from instruction in subgoals formation strategies. No differences were found between groups, indicating that the type of practice problem given had no effect on strategy development.

STUDY TWO

Subjects. Subjects were 100 fourth through sixth grade students at the same private elementary school. All had at least 30 hours previous Logo programming experience.

Methodology. All subjects were tested on their ability to apply the five problem solving strategies on which subjects improved in the first study -- subgoals formation, forward chaining, systematic trial and error, alternative representation, and analogy, and randomly assigned by grade to one of three treatment groups receiving respectively the explicit training/programming practice intervention, explicit training with cut paper manipulatives practice, or a Logo discovery learning experience with no strategy training. On completion of all treatments, subjects were post-tested using different but analogous tests. Differences between mean pre- and post-test scores were examined using a three-way analysis of variance. Independent variables were test, strategy, and group. The dependent variables were scores on the strategies tests.

Results. Significant differences in pre- to post-test increases were found between groups. Further analysis of this finding revealed that the explicit training/Logo programming group, and that group alone, showed significantly improved subgoals formation, forward chaining, systematic trial and error, and analogy strategies. Increased ability in applying alternative representation strategies was indicated but not conclusively demonstrated. The results thus argue for the superiority of explicit training and programming practice over both similar instruction with cut-paper manipulatives practice and discovery learning in a similar practice environment for the teaching and learning of problem solving.

STUDY THREE

Subjects. Subjects were 40 eleventh and twelfth grade students at an American school in Switzerland enrolled in one of three classes -- a Logo class, an AP Pascal class, or a Pre-Calculus class. No subjects had any previous Logo programming experience.

Methodology. All subjects were given paper-and-pencil tests of their ability to apply the five problem solving strategies on which subjects improved in the first study. Subjects in the Logo class received explicit instruction and Logo programming practice in each strategy. Subjects in the AP Pascal and Pre-Calculus classes received regular content area instruction. On completion of all treatments, subjects were post-tested using different but analogous tests. Differences between mean pre- and post-test scores were examined using a three-way analysis of variance. Independent variables were test, strategy, and group. The dependent variables were the scores on the tests of each of the problem solving strategies.

Results. Significant differences in pre- to post-test increases were found between groups. Further analysis of this finding revealed that subjects in the Logo class showed significantly improved subgoals formation, forward chaining, and systematic trial and error strategies. Increased ability in applying alternative representation and analogy strategies was also indicated but not conclusively demonstrated for this group. The results again argue for the superiority of explicit strategy training and Logo programming practice over regular instruction in subjects traditionally prescribed for the teaching and learning of problem solving and demonstrate the efficacy of the instructional model we developed with a very different student population.

STUDY FOUR

Subjects. Subjects for this study were 28 graduate students of education taking Logo programming courses at the State University of New York at Albany.

Methodology. All subjects were pretested on their ability to apply the five problem solving strategies. Subjects in one class were taught problem solving along with Logo programming; subjects in the other class were taught Logo programming alone. At the end of the semester, subjects were post-tested using different but analogous tests. Differences between mean pre- and post-test scores were examined using a three-way analysis of variance. Independent variables were test, strategy, and group. The dependent variables were scores on the tests of each of the problem solving strategies.

Results. Significant differences in pre- to post-test increases were found between groups. Further analysis of this finding revealed that subjects in the Logo/problem solving class demonstrated significantly greater improvements in their ability to apply subgoals formation, forward chaining, alternative representation, and analogy strategies than did students who were taught Logo programming alone. Students in both classes improved in their ability to apply subgoals formation strategies. The results replicate those of the previous studies for in-service teachers. They further thus that we must teach teachers problem solving strategies if we would have them, in turn, teach such strategies to their students.

DISCUSSION

The results of these studies demonstrate that five problem solving strategies -- subgoals formation, forward chaining, systematic trial and error, alternative representation, and analogy -- can be developed in students explicitly taught those strategies and given practice applying them to solve Logo programming problems; that such intervention is more effective than Logo discovery learning, explicit instruction with concrete manipulatives practice, and instruction in subjects traditionally prescribed for the teaching and learning of problem solving. The findings thus highlight two features of the intervention that seem integral to its success -- knowledge-based instructional design and the Logo programming environment itself.

Knowledge-based instructional design refers to premising the design of instruction on desired knowledge outcomes rather than on desired behavioral outcomes. The distinction is a real one. The desired outcome of problem solving instruction, for example, is increased problem solving abilities. When these abilities are conceived as behavioral manifestations, the prescribed instruction has involved practice solving relevant problems. Studies two, three, and four demonstrate that practice alone does not result in increased problem solving abilities. When these abilities are conceived in terms of knowledge outcomes, however, the focus is not on problem solving behaviors, but on the knowledge supporting such behaviors. In the instruction we designed, problem solving strategies were broken into their component steps and explicitly taught. Students were thus provided with declarative knowledge of the problem solving strategies to be learned.

Declarative knowledge of particular problem solving strategies, however, is not in itself enough to ensure their development, as shown by study two. Procedural knowledge is also necessary. It is our belief that programming environments, the Logo environment in particular, are uniquely conducive to the development of such skills because they support quasi-concrete representations of these abstract strategies that students can inspect, manipulate, and test through practice. In this vein, it is instructive that alternative representation strategies, for which their are no direct Logo representation, were the least likely to be developed by the students in our studies, while subgoals formation strategies, which are the most concretely represented in the language, were most likely to be developed.

REFERENCES

1. S. Papert, Mindstorms (Basic Books, NY, 1980)
2. D. Watt, Byte, 7(8), 116, (1982)
3. R. D. Pea & D. M. Kurland, New Ideas in Psych., 2 (2), 137, (1984)
4. U. Leron, The Computing Teacher, 12 (6), 26, (1985)
5. D. G. Shaw, AEDS Jl., 19 (2/3), 176, (1986)
6. R. P. Johanson, Jl. of Ed. Comp. Research, 4 (1), 1, (1988)
7. G. Salomon & D. Perkins, Jl. of Ed. Comp. Research, 3 (2), 149, (1987)
8. S. M. Carver, Comp.Systems Group Bul., 14 (1), 4, (1987)
9. D. H. Clements, Jl. of Ed. Comp. Research, 3 (1), 94, (1987)
10. R. Lehrer, L. Sancilio & L. Randle, Learning pre-proof geometry with Logo. AERA, New Orleans, (1988)
11. A. D. Thompson & H. M. C. Wang, Jl. of Ed. Comp. Research, 4 (3), 335, (1988)