Programming and Problem Solving: What Have We Learned?
Demetria Ennis

Problem solving and programming have received a great deal of attention over the years. Many individuals have claimed that computer programming teaches problem solving and reasoning and helps students learn about their thought processes. This claim has not been proven, nor is it new. Mathematics instruction (geometry and trigonometry), The Latin School Movement of the 1600s, Logic instruction, and Writing Systems were introduced in classes with the same claims.

Papert is one of the most recent and most vocal individuals to make this claim. Papert claims that Logo, a language for learning, teaches students how to think and helps them learn about their thought processes. It has not been demonstrated that Logo fulfills Papert’s promise, but programming in Logo and other languages does provide students with a useful arena to practice and develop some of their cognitive skills.

This article will summarize some of the programming and problem solving research conducted from 1983 to 1990. It will conclude with a summation of what we’ve learned from programming and problem solving research.

Does Programming Instruction Have Any Cognitive Benefits?

Programming instruction has many specific cognitive benefits. It can teach procedural skills, general programming concepts, and general problem solving algorithms. We’ve learned (through research) that programming instruction can teach students procedural skills (Linn, 1985; Pea & Kurland, 1984; Dalbey & Linn, 1985; Mayer, Dyck & Vilberg, 1986). Procedural skills are planning, decomposing, testing, and redesigning. Planning relies on organizational skills and understanding the problem. Once the problem is understood, it can be broken into smaller parts or subroutines; this is decomposition. Testing and redesigning both involve restructuring a problem or parts of a problem in order to design a correct solution.

Programming instruction allows students to acquire information that can be used in any programming language. General concepts like looping, pseudocode development, data structures, input and output strategies and basic problem solving algorithms can be learned and applied to any programming language. When programming instruction focuses on generalizable concepts it fosters transfer, and it encourages students to embrace what they’ve learned and make it a part of their knowledge repertoire.

Generalized problem solving algorithms are another cognitive benefit of programming. These algorithms are general programming subroutines that perform basic functions like sorting or searching. Students who have access to general programming routines can learn how to use these routines to solve more complicated problems (Linn, 1985; Dalbey & Linn, 1985; Mayer, Dyck & Vilberg, 1986). They can also learn to translate them into different programming languages. They can also use these routines to explore new techniques and complete new projects in shorter time periods.

Research On Programming and Problem Solving

A great deal of research has been done on programming and problem solving. The computer programming language most studied has been Logo. Many researchers have examined Logo and its ability to teach students how to solve problems and think. Some studies have also been conducted using BASIC. Most educators express concern about using BASIC. They don’t feel that it handles data structures well, encourages problem decomposition, or structured programming. Still, it continues to be taught because of its ease of use and availability.
Logo has been more widely studied because its creator, Seymour Papert, suggested that it would teach problem solving and the thinking process in a child-centered, self-discovery learning environment. Papert also claimed that Logo would support children by allowing them to build their own intellectual models, apply these models to other situations, and think about this entire process. The discovery learning programming environment Papert espoused was designed to allow students to interact and participate while they followed their own self-directed path. This type of environment failed to produce the desired effects (Palumbo, 1990; Gavri, 1988; Collis, 1988; Swan, 1989; Lieberman, 1988; Lehrer & Smith, 1988). However, with the proper instructional techniques, Logo programming instruction can be beneficial.

BASIC (Beginner's All-Purpose Symbolic Instruction Code) was developed by Kemeny and Kurtz in the mid 1960s as an educational tool. BASIC has a simple algebraic structure and is interpreted or directly executed rather than compiled like most high level languages. BASIC's popularity can be credited to its small size, easy of use, fast execution, interpretive nature, the advent of the microcomputer and free BASIC software.

The claims that programming teaches problem solving or improves problem solving ability have been examined by many researchers. The results of these studies are mixed. There are some studies which show promise, but most of the research indicates that programming instruction alone doesn't teach problem solving skills. This research also suggests that improvements in problem solving are directly related to the concepts emphasized in the programming language, the instruction provided and the instructional techniques used (Clement, Kurland, Mawby & Pea, 1986; Milojkovic, 1983; McCoy & Dodell, 1989; Lehrer & Smith, 1988; Fay & Mayer, 1988).

What We've Learned From Programming and Problem Solving Research

In 1983 Roy D. Pea suggested that problem solving or thinking skills should be taught directly. Pea suggested that explicit instruction in problem solving was more effective than assuming that these skills would occur as a by-product of programming instruction. Pea found that Logo was "cognitively complex beyond its early steps, and quite difficult to learn without instructional guidance" (Pea, 1983, p. 2). Pea made this conclusion after extensive studies of Logo with 8-9 and 11-12 year olds. He indicated that young Logo students learned the semantics and syntax of the language easily, but they didn't learn how to structure and organize programming statements in order to accomplish a specific goal.

Pea, Kurland, and Hawkins (1983), identified planning as an essential problem solving and thinking tool. They performed an experimental study on the planning skills of students (8-9 & 11-12 years old) after these students had one year of Logo programming. They found that Logo programming did not have a measurable influence on the planning abilities of these students. They concluded that learning to think and plan were not intrinsically guaranteed by Logo. In order to teach or improve a student's problem solving skills through programming many researchers have stressed the need for a structured curriculum. This curriculum should emphasize the skills to be learned and encourage children to deliberately look for connections with material outside the programming domain (Gavri, 1988; Littlefield, Delclos, Lever, Clayton, Bransford, Franks, 1988; Seidman, 1989).

In 1984 Clements and Gullo found that their Logo treatment group improved in language fluency, divergent and reflective thinking, and the ability to give directions. This study is one of the few that supports the idea that Logo programming instruction improved certain problem solving skills.

Shaw (1984), conducted a study to examine whether learning to program in BASIC or Logo was more effective in developing problem solving skills as opposed to no programming instruction. Shaw had experimental and control groups of fifth grade students, and she compared their pretest and posttest scores on the New Jersey Test of Reasoning Skills. She found no significant difference in the posttest scores of the BASIC and Logo groups. She concluded that neither language sufficiently developed problem solving skills.

Kurshan and Williams (1985), conducted a study to determine whether microcomputer use increased the problem solving abilities of junior high school students. One group was exposed to introductory computer literacy and simple computer programming while the other group lacked this experience. Problem solving performance was measured by the problem solving subtest of the SRA Achievement Series Test. The researchers found that boys, but not girls, improved their problem solving scores. They also reported that girls had less exposure to computer courses and computerized games, and girls' problem solving gain scores were significantly lower than boys.

In 1986 Mann compared the problem solving abilities of eighth grade students instructed in Logo as
compared to those who received no instruction in Logo. He used the Cognitive Abilities Test (CAT): Non Verbal Battery. After a special Logo programming class was conducted daily for ten weeks, Mann compared the groups and found that the Logo group's problem solving abilities were enhanced as a result of learning Logo programming skills. Gender was not a factor in this improvement.

In 1986 Horton and Ryba reported positive results from a study they conducted. Their Logo and non-programming groups were assessed on six tasks: exploration, analysis and planning, creativity, debugging, coding, and prediction. After administering their treatment, both groups received a posttest on the six tasks mentioned earlier. The researchers found that the Logo group outperformed their non-Logo group on five of the tasks: exploration, analysis and planning, coding, creativity, and prediction. Although small sample sizes were used, the students in the Logo group seemed to improve their ability to write directions, estimate designs and perform block designing tasks.

Collis reviewed twelve Logo studies from 1985-89 and found two trends: (1) the teacher and the level of instructional support used with Logo were factors that strongly influenced the impact of Logo on students; (2) skills gained within the context of using Logo were not easily or often transferred to mathematical problems or other cognitive areas. Collis indicated that Logo's ability to enhance thinking skills or problem solving is heavily influenced by the instructional guidance the teacher gives the student within the Logo environment. Collis concluded her review by saying that "Logo is a good tool, but not one which most students will use to much advantage on their own without thoughtful guidance" (Collis, 1988, p. 15).

Swan and Black (1988), found significant differences between the pretest and posttest scores of eighth grade students on tests involving six different problem solving strategies: subgoal formation, forward chaining, backward chaining, systematic trial and error, alternative representation, and analogy. The results supported their hypothesis that a combination of general problem solving strategies, direct instruction and a mediated learning environment enabled students to develop transference problem solving skills from their Logo experience.

Vanlengen 1988, attempted to discover whether programming instruction improved the problem solving ability of college level introductory computer students. His hypothesis was that college level students who mastered computer programming would score higher on the Watson - Glaser Critical Thinking Appraisal than students who received instruction in the use of only spreadsheet and database management software. Mastery was determined by how well students performed on examinations. To be included in the study, students had to attain a score indicating 75% or greater mastery. Vanlengen found that there was no significant increase in the problem solving ability of either group. The author suggested that instruction in problem solving strategies, guided, mediated instruction, and longer treatment periods might produce increases in problem solving performance.

Swan 1989, conducted a study to determine whether an instructional model that combined mediated practice, discovery learning, and direct instruction using Logo resulted in transfer or increased problem solving abilities. Swan reexamined the five problem solving strategies she researched in an earlier study with J. B. Black. In this study, Swan tried to determine whether transfer occurred. Her subjects were 100 fourth through sixth grade students who had at least one year of prior Logo programming experience. She found transfer of problem solving skills in four of the problem solving strategies she investigated—analogy, systematic trial and error, forward chaining, and subgoal formation. Swan's concluded that Logo could be used successfully as a medium to teach and learn problem solving skills.

According to Fay and Mayer (1988), programming instruction can help students learn general thinking skills, but in order for this to occur these thinking skills should be demonstrated, learned, and practiced in the programming environment. After this type of direct instruction, the student must recognize that these skills can be applied to other areas outside the programming domain. Fay and Mayer drew this conclusion after conducting a study with 30 fourth grade students to determine whether cognitive changes occurred when fourth graders learned the syntax and semantics of Logo.

Lieberman (1988), reviewed studies and articles on Logo and cognitive skill development. She concluded that older children and those with previous computer experiences gained more from Logo instruction than younger or less experienced children. She also indicated that the effectiveness of Logo instruction was directly related to the teacher's role and the instructional methods used.

In 1986 Lehrer and Smith examined Logo instruction using classes of third grade students. One group of students got direct teacher guidance, and they were encouraged to practice the new skills they
learned. The other group of students was left to
discover Logo on its own. The group that received
direct Logo instruction was able to demonstrate
transferable thinking skills; the self-discovery group
was unable to demonstrate these skills.

According to a study by Mayer, Dyck and Vilberg
(1986), learning a programming language can change
thinking skills, but improvements in thinking skills
seem to be directly related to concepts emphasized
within the language. Other factors that can change
thinking skills are explicit instruction, guided prac-
tice and a structured programming curriculum.

Summary

Programming instruction can be beneficial for stu-
dents. It can teach procedural skills, general pro-
gramming concepts, and algorithms (Linn, 1985;
Dalbey & Linn, 1985; Mayer, Dyck, & Vilberg, 1986).
After years of research on programming instruction
we've learned the following things:

• programming instruction does have specific cog-
nitive benefits (procedural skill acquisition, gen-
eral concepts, and general problem solving algo-
rithms) that can help a student improve his or her
problem solving and thinking skills (Linn, 1985;
Dalbey & Linn, 1985);

• direct guidance and good instructional tech-
niques for programming and problem solving are
essential if students are going to learn useful
and transferable problem solving skills (Govier,
1988; Lehrer & Smith, 1988; Lieberman, 1988,
Swan, 1989);

• it's necessary for instructors to emphasize the
problem solving skills that students need to learn
(Clement, Kurland, Mawby & Pea, 1986;
Milojkovic, 1983; Fay & Mayer, 1988);

• no programming language intrinsically teaches
or enhances a student's problem solving ability
(Govier, 1988; Littlefield, Delcols, Lever, Clayton,
Bransford, Franks, 1988; Seidman, 1987);

• problem solving skills learned in a programming
environment are not automatically or easily trans-
ferred to another domain (Collis, 1989; Govier,
1988; McCoy & Dodl, 1989; Lehrer, Guckenber
& Sancillo, 1988),

• programming instruction can facilitate problem
solving or thinking abilities, but this does not

occur without specific problem solving instruc-
tion, direct teacher guidance, mediated practice
(Clement, Kurland, Mawby & Pea, 1986; Swan,
1989, Salomon & Perkins, 1989, Palumbo, 1990);
and

• instruction which encourages students to look for
relationships between dissimilar information and
apply the skills they've learned is essential.

Demetria Ennis is a doctoral candidate in computer-
based education at Kansas State University, 261
Bluemont Hall, Manhattan, Kansas, 66506.