THE EFFECTS OF LEARNING TO PROGRAM A COMPUTER
IN BASIC OR LOGO ON THE PROBLEM-SOLVING
ABILITIES OF FIFTH GRADE STUDENTS

DISSERTATION

Presented to the Graduate Council of the
North Texas State University in Partial
Fulfillment of the Requirements

For the Degree of

DOCTOR OF PHILOSOPHY

By

Donna Gail Shaw, B.S., M.Ed.
Denton, Texas
August, 1984
Shaw, D. G. *The Effects of Learning to Program a Computer in BASIC or Logo on the Problem-Solving Abilities of Fifth Grade Students*. Doctor of Philosophy (Elementary Education), August, 1984, 96 pp., 5 tables, 74 titles.

The purposes of this study were (1) to determine if learning to program a computer in either BASIC or Logo improves the problem-solving skills of fifth grade students when compared to a control group that receives no programming instruction, and (2) to determine if learning to program a computer in Logo is more effective than learning to program in BASIC for improving problem-solving skills in fifth grade students. Subjects were 132 fifth graders from two suburban elementary schools. The materials used in the study were the *Computer Challenge Guide* for the BASIC group and *Logo in the Classroom* for the Logo group. The *New Jersey Test of Reasoning Skills* was used as the pretest and posttest measure.

A computer-usage questionnaire, developed by the researcher, was administered to the subjects prior to the study. Any students indicating the ability to program a computer in either BASIC or Logo were eliminated from the final data analysis. The pretest measure was given followed by a seven-week treatment period. Students in the BASIC experimental group received one hour of computer instruction
per day in BASIC programming. Students in the Logo experimental group received the same amount of instruction time in Logo programming. The control group received no programming instruction at school. The posttest measure was administered at the end of the treatment period and the results were analyzed with the analysis of covariance procedure.

The hypotheses of the study predicted that (1) the students in the programming groups would achieve significantly higher adjusted posttest scores than the students in the control group, and (2) that the students in the Logo programming group would achieve significantly higher posttest scores than the students in the BASIC programming group. Results indicated that learning to program in BASIC or Logo does not significantly improve problem-solving ability as measured by the New Jersey Test of Reasoning Skills when compared to a group that received no programming instruction.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>II. SYNTHESIS OF RELATED LITERATURE</td>
<td>7</td>
</tr>
<tr>
<td>III. METHODS AND PROCEDURES</td>
<td>30</td>
</tr>
<tr>
<td>IV. RESULTS</td>
<td>45</td>
</tr>
<tr>
<td>V. DISCUSSION</td>
<td>51</td>
</tr>
</tbody>
</table>

**LIST OF TABLES**

||
|---|---|
| vi |
APPENDIXES ................................................. 61
BIBLIOGRAPHY .............................................. 90
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I.</td>
<td>Demographic Data</td>
<td>45</td>
</tr>
<tr>
<td>II.</td>
<td>Student Access to Computers</td>
<td>46</td>
</tr>
<tr>
<td>III.</td>
<td>Subject Attitudes toward Computers</td>
<td>47</td>
</tr>
<tr>
<td>IV.</td>
<td>Analysis of Covariance for the Posttest</td>
<td>49</td>
</tr>
<tr>
<td>V.</td>
<td>Gains in Problem Solving from Pretest to Posttest</td>
<td>50</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

Background

Knowledge of facts alone does not solve problems. Though factual knowledge is important, children need instruction in organizing thoughts, reflecting upon situations and comprehending the meaning of events. Teaching a child how to think is an important process in developing his intelligence (11, pp. 38, 40).

The utilization of microcomputers may play a central role in helping children become better problem solvers. Research reveals initial evidence that the same processes are used in both computer programming and problem solving (10). Consequently, computer programming may be an effective way of enhancing the problem-solving process. Teaching these skills at an early age is important (1, 9).

There are several languages available for teaching programming to children. Two of the most popular languages are BASIC and Logo. BASIC is the programming language currently being taught in most educational settings. Research has shown that learning to program the computer in BASIC can improve problem-solving abilities (8).
Papert and his colleagues at MIT, the developers of Logo, had as their primary concern the development of a computer language that would help children become better learners. They believed that in order to become better learners, students needed to develop good strategies for solving problems. Logo was designed to facilitate the development of these problem-solving strategies (2, p. 37). Field tests indicate that Logo does facilitate problem-solving abilities (6).

Since factual knowledge is relatively useless without the processes for operating on that knowledge to solve problems and answer questions, it is imperative that an effective means for developing problem-solving strategies be found. This study was designed to investigate the relative effects of learning to program a computer in Logo or BASIC on problem-solving abilities.

Statement of the Problem

The problem of this study was a comparison of the effects of two programming languages on problem-solving skills.

Purpose of the Study

The purposes of this study were

1. To determine if learning to program in either BASIC or Logo improves problem-solving skills of fifth grade students.
2. To determine if learning to program in Logo is more effective than learning to program in BASIC for improving problem-solving skills in fifth grade students.

Hypotheses

In order to carry out the purpose of the study, the following hypotheses were tested.

1. Students in the BASIC programming group will achieve significantly higher posttest scores on the **New Jersey Test of Reasoning Skills** than will students in the control group.

2. Students in the Logo programming group will achieve significantly higher posttest scores on the **New Jersey Test of Reasoning Skills** than will students in the control group.

3. Students in the Logo programming group will achieve significantly higher posttest scores on the **New Jersey Test of Reasoning Skills** than will students in the BASIC programming group.

Significance of the Study

Schools have been charged with the responsibility of developing thinking skills. It has been stated that "no topic that is presented to children in school is more essential and more valuable than problem solving" (5, p. 26). Papert (7) believes that learning to program a computer can improve one's ability to think. Logo is being accepted by many schools throughout the country as a powerful tool for developing thinking skills, yet the resident language of the...
microcomputer is BASIC. This study was significant in that it examined the effectiveness of learning computer pro-
gramming in enhancing problem-solving abilities and provided information that could be valuable in making curriculum decisions regarding computer usage in the elementary school.

Definition of Terms

The following terms have restricted meaning for this study

**Problem solving** is the process by which the learner discovers a combination of previously learned rules that he can apply to achieve a solution for a novel problem situation (4, p. 214). Problem-solving ability is measured by scores obtained on the New Jersey Test of Reasoning Skills in this study. The terms problem solving, thinking and reasoning are used interchangeably for purposes of this study.

**Transfer of learning** means that experience or performance on one task influences performance on some subsequent task (3, p. 3). Transfer of learning and transfer of training are used interchangeably for purposes of this study.

**Programming** refers to the demonstrated ability to utilize the syntax of a computer language to produce desired results.

**Logo** is the name of the computer language developed by Seymour Papert and associates at MIT.
Primitives are the "built-in" words of the Logo language.

BASIC (Beginner's All-purpose Symbolic Instruction Code) is an acronym for the computer language developed at Dartmouth; it is the resident language in most microcomputers.

Limitations

The study had two limitations which must be recognized. One limitation concerned the possible teacher effects in that one teacher may have been a better instructor than the other. A second limitation was the student effects. Both the Logo and BASIC programming groups were in the same school. It was beyond the researcher's control to prevent the possible communication and after school collaboration of students from different programming groups. In addition, by necessity the control group and the experimental group were in separate schools.

Basic Assumption

It was assumed that the cooperating teachers would follow the sequence and structure of the materials used in this study.
CHAPTER BIBLIOGRAPHY


CHAPTER II

SYNTHESIS OF RELATED LITERATURE

Though research evidence tends to support the contention that learning to program in Logo may develop problem-solving skills, a thorough review of the literature revealed no effort to determine if learning to program in Logo is more effective than learning to program in BASIC for improving problem-solving skills in fifth grade students. Several areas of research that are related to the present investigation are reviewed below.

Problem Solving and Transfer of Learning

One of the anticipated gains of the use of microcomputers in the home and school is an increase in problem-solving abilities and the subsequent transfer of these abilities to general problem-solving skills. Therefore, research investigating various aspects of problem solving and transfer of learning are considered pertinent to this study.

In the past, the major focus of the mathematics curriculum has been computation; however, technology has progressed to the point that computation alone is not a marketable skill (45). The National Council of Supervisors of Mathematics claimed problem solving to be the number-one
basic skill in its "Position Paper on Basic Mathematics Skills" (34). In like manner, the National Council of Teachers of Mathematics in its "Agenda for Action" (35) stated that problem solving should be the focus of the curriculum for the 1980s. According to Slesnick (45), though the teaching of problem solving is usually entrusted to the mathematics curriculum, it should not be limited to the mathematics classroom. Problem solving is of such importance that it should be a part of every subject taught in school.

Problem solving is commonly used in a variety of contexts (43). It may be used for responding to complex situations, doing homework exercises, solving puzzles, creating inventions, or resolving interpersonal conflicts. Planning, analyzing, creating, evaluating, persuading or detecting may all be referred to as problem-solving activities.

According to Shann (43), the concept of problem solving has been derived mainly through logical analysis. Psychologists theorizing on problem solving have postulated models of the process which appear to follow one of two approaches: "(1) the identification of subgroups of intellectual processes linked in some order, whose linkages may be of the linear or feedback variety; and (2) the consideration of problem solving as the operation of an information processing system" (43, p. 12).
Those theorists and psychologists whose models identify component intellectual processes reflect the influence of Dewey. Dewey lists five logical steps in the problem-solving process:

1. a felt difficulty
2. its location
3. suggestion of possible solution
4. development by reasoning of the bearings of the suggestions
5. further observation and experiment leading to its acceptance or rejection, that is the conclusion of belief or disbelief (10, pp. 72-77)

The more current models in this category all resemble the phases proposed by Dewey. For example, Polya (39), a mathematician, listed four steps in the problem-solving process: understanding the problem; working out connections between the known and unknown, thus deriving a plan of solution; carrying out the plan; and examining the solution. Gagne' (14) also lists four steps involved in problem solving: presentation of the problem, defining the problem, formulating hypotheses, and verifying the hypotheses. Likewise, Krulick and Rudnick (22) list four steps in the heuristics of problem solving: understanding the problem, exploring or formulating hypotheses and developing a strategy to test each hypothesis, carrying out the strategy, and checking the results.
Other models of problem solving might be described as information processing models. The TOTE feedback system presented by Miller, Galanter, and Pribram (31) is one such model. This model involves testing a given condition; if the condition is not satisfied, performing some one of a small class of operators and testing the condition again, and finally exiting when the criterion is met. The TOTE model does not specify any particular abilities in problem solving but lists the form of the processes: test, operate, test, exit. Guilford (18) proposed a more elaborate information processing model with potential exit points and access to memory storage after each phase. In addition, Newell and Simon (36) derived an information processing model of human problem solving based on their work on computer simulation of human reasoning.

A common theme in research on thinking is the need to be exposed to instruction in problem solving or the need to be taught a strategy to solve problems. Krull (23) states that

No topic that is presented to children in school is more essential and more valuable than problem solving. Problem solving transcends the traditional subject-matter lines by requiring a facility with language and a general understanding of the world. For indeed, the problems that our students will face when they leave school will probably differ significantly from traditional textbook word problems. Since we cannot hope to predict what kinds of problems will face these students in the twenty-first century, the skills of problem solving must be general and applicable to a wide variety of situations (p. 26).
Several studies (3, 4, 25, 33, 40, 42) indicate that specific problem-solving skills, such as analytical thinking, mathematical problem solving, interpersonal problem solving, productive thinking, and conditional reasoning, improve following instruction in that particular skill. Studies by Moran (32), Clarke (6) and Anderson (1) emphasize the use of strategy training for more effective and efficient means of problem solving. In particular, Goldberg (15), after studying problem-solving abilities of selected seventh graders, suggests that strategy games can be effective in improving problem-solving skills. Krulick and Rudnick (23) support this contention since the heuristics of problem solving and strategy gaming are similar.

Another common theme in research on thinking is the need to be aware of one's own thinking process. Piaget (38) and Papert (37) contend that self-awareness is necessary for self-correction. Whimbey and Lochhead (53) suggest that one approach to encourage children to become more aware of their thinking is to have students work in pairs. One student works on a problem talking out loud. The other student, by listening, follows each step in the first student's reasoning. The objective is to have each student learn to listen to his own thoughts (26). Similarly, when students debug their computer programs, they are studying their own thinking (37).
As evidenced by the numerous studies conducted, problem solving is a major concern of educational research. Also of great concern is the idea of transfer of learning or training. Many authorities (9, 12, 44) contend that transfer of learning is the most important topic in the psychology of learning. Bamvakais (2) has had some success in the utilization of an instructional program based on a cognitive information processing theory of learning that enhances the transfer of critical thinking skills. However, Bredie (4), in a study involving eighty-one sixth grade students, found that the acquired heuristics in word problems did not transfer to verbal reasoning problems as measured by a verbal reasoning transfer test. In addition, Clarke (6), in studying kindergarten and first grade children, found that the transfer of training to a highly similar problem-solving task previously performed was limited as determined by a matrix solution task and a twenty questions task.

Transfer of learning and problem solving have been concerns of researchers and educators for many years. The recent development of the microcomputer has brought about a renewed interest in these ideas as well as a new area of research focusing on the use of microcomputers as a means for developing problem-solving skills (23). Holzman and Glaser (21) suggest that some interesting facts about children's problem-solving techniques and learning processes might be
acquired by studying the kinds of programs children write and the types of programming concepts with which they have difficulty.

Effects of the Computer on Problem Solving and Transfer of Learning

According to a recent survey published by Johns Hopkins University (49), computers are mostly used in elementary schools for computer literacy and drill and practice. A review of the literature reveals that much of the research dealing with the computer and learning has focused on computer assisted instruction, with only a few studies concentrating on programming the computer. Most of these studies address the research areas of problem solving and transfer of learning.

It has been predicted that the enhancement of problem-solving abilities is one of several ways in which microcomputers will dramatically change the intellectual development of children (48). Current research seems to support this prediction. Cox (7) found that seventh and eighth grade students improved their problem-solving ability in a short time by participating in three interactive problem-solving microcomputer programs dealing with topics from life science, social studies and environmental education. Data for subject identification, time and clues needed for solution, and order of clues were saved using the microcomputer for later retrieval. Anecdotal observations of
subjects with other subjects and with microcomputers were recorded. Program difficulty, interactions, problem-solving strategies and achievement over the three program sessions were analyzed. Similarly, Elg (13) investigated the effect of computer simulation experience on fifth graders' problem-solving ability and found that the treatment significantly increased their problem-solving abilities.

Other studies concerned with developing problem-solving skills have not found evidence to support the hypothesis that use of the computer enhances problem-solving skills. Mandelbaum (28) found that utilization of the computer as a problem-solving tool with low performing tenth grade students did not improve their ability to solve problems involving computation, concepts and applications as measured by selected sections from the Comprehensive Test of Basic Skills. In addition, vonStein (50) found that the utilization of the computer to help kindergarten students learn patterning and counting shapes was no more effective than small group instruction.

While studies concerned with the utilization of computer assisted instruction in developing problem-solving skills continue to be conducted, a new area of research is rapidly gaining popularity. This new area of study deals with learning to program a computer as it relates to problem solving and transfer of learning. Statz's (47) research, in which a group of children ranging in age from nine to eleven
years were taught to program in Logo, reveals that initial results with transfer of general problem-solving abilities from programming experiences are tenuous. This was determined through the use of a battery of pre- and post-tasks. However, more recent studies are producing different results. Wells (52) conducted an exploratory study of the relationship between the processes involved in problem solving and the processes involved in computer programming. Her findings suggest that the same processes are used in computer programming and in problem solving. In a study conducted by Rose (41), forty fifth graders were given approximately twenty hours of programming instruction in BASIC. It was concluded that computer programming instruction seemed to improve the logical problem-solving abilities of fifth graders and that the findings suggest a transfer of computer programming skills to general problem-solving ability as measured by the Cornell Critical Thinking Test and the Developing Cognitive Abilities Test. Consequently, learning computer programming may be an effective way of enhancing the problem-solving process. These studies, plus other research evidence compiled by D'Angelo (cited in 54), suggests that problem-solving abilities developed within a programming context may transfer to problem-solving abilities within other contexts.
Computer Programming and Problem Solving

According to Cannara (5), three main streams of thought center round the study of children learning to use a computer. First is the view that a computer is a tool for thinking or, more specifically, an instrument used to stimulate the activity of "thinking about thinking" (37). Second is the inclination of some educators to study the thinking processes of people solving problems which, according to Cannara (5), leads to studies in computer programming. Third is the area of artificial intelligence research which aims to formalize problem-solving procedures. All three streams of thought deal with some aspect of thinking or problem solving.

Since the development of Logo by Seymour Papert and his associates, there has been much interest among educators and researchers concerning the utilization of the computer as an instrument for improving the thinking process. This interest has manifested itself in such computer programming projects as the Edinburgh Logo Project, the Brookline Logo Project, the Computers in Schools Project and the Lamplighter School Logo Project (51). The alleged gain from learning to program a computer is an improvement in logical thinking. According to Gorman, "although such gains are assumed almost as a cultural truism, there is little research to either support or inform a gain" (17, p. 165).
The contention that learning to program aids problem solving implies that any computer language would prove satisfactory in this endeavor. Yet there is a continuous debate among programmers over which language is best suited for this particular application (16). According to Holzman and Glaser (21), the most important criterion for selecting a programming language for children is the extent to which its words are similar to conversational English. Both BASIC and Logo meet this criterion.

BASIC, according to Gorman (16) and Solomon (46), was designed to be an easily learned computer language. BASIC code was limited to a few commands in order to enhance the ease of learning a first computer language, and it is the most common language available on the microcomputer. Problem-solving learning was not a major consideration in the designing of the BASIC language. However, this does not mean that BASIC could not be effective for this type of learning (16).

The literature reveals that little research has been conducted concerning the effects of learning to program in BASIC on problem-solving abilities. The few studies that have been conducted are positive. Holzman and Glaser (21) studied six eleven-year-old boys as they learned to program in FOCAL (a language similar to BASIC) or Logo. The students' problem-solving techniques and learning processes were studied. According to the researchers, the students became
competent in problem solving on the computer, and they con-
cluded that both FOCAL and Logo were appropriate languages
for sixth graders to learn. Similarly, Doorly (11) reported
that gifted children from six to ten years old were success-
ful in problem solving or programming in BASIC. She noted
that "these learners not only possess the ability to apply
the necessary logical thinking and mathematical skills to
operate the computer but also understand the mechanics of
what is taking place" (p. 64). In another study, Krull (24)
found that elementary students' mathematical problem-solving
ability was significantly improved after students were en-
gaged in writing BASIC computer programs which focused on
several properties in mathematics. Pre- and post-tests in
math achievement and attitude surveys were administered to
test the effectiveness of the programming activities. Most
recently, Rose (41) reported that computer programming in
BASIC can enhance the problem-solving abilities of fifth
graders in specific areas such as hypothesis testing and
determination of the relevance of data.

Logo, like BASIC, was designed to make computer pro-
gramming as easy as possible. According to Hines (20),
five-year old children can successfully program in Logo. It
was designed as a language for learning in general—a lan-
guage for learning how to think (19, 37). Logo's history is
strongly rooted in artificial intelligence as well as in
Piaget's research into how children develop thinking skills
Because of its design, many educators and computer scientists contend that Logo encourages the development of problem-solving strategies (19, 30, 37, 46). Research in the development of computer programming concepts and problem-solving abilities among ten year olds learning Logo, conducted by Statz (47), failed to support this contention. However, more recent research appears to be more supportive of the original intent of the programming language.

A two-year project, known as the Edinburgh Logo Project, was designed to discover whether the ability to do and talk about mathematics is changed after exploring mathematical problems through Logo programming. The project involved eleven sixth grade boys from a private school near the University of Edinburgh. These sixth graders were selected from the school's lowest level math group. The students attended a Logo lab at the university. The first year, the students worked through a set of graded worksheets to learn the basic elements of Logo. The second year, the students did special Logo exercises designed to teach topics from their regular mathematics curriculum. Pre- and post-standardized math tests were given to the experimental and control group. The experimental group improved slightly more than the control group on a basic math test though this improvement was not statistically significant. However, the reverse was true on a math attainment test. According to Watt (51), the most interesting finding had to do with the
teachers' perceptions of the students in both groups. The Logo classes were more willing to "argue sensibly about mathematical issues" and to explain their "mathematical difficulties clearly" (51, p. 118).

In 1977, at Lincoln High School near Brookline, Massachusetts, a computer laboratory was set up for sixth graders to learn Logo. Fifty sixth graders were given the opportunity to learn Logo. The work of sixteen of these students, ranging from academically gifted to learning disabled, was selected for study. The entire Logo experience of these students was monitored and analyzed. What the students learned, the learning styles which they used and the types of choices which they made were carefully documented. Standardized tests were not used because they were considered irrelevant to the goals of the project. The problem-solving tests and mathematics tests devised and administered by the project staff showed inconclusive results. However, the results of the project indicated that Logo learning environments are suitable for different kinds of students and all students were successful in Logo problem solving or programming (51).

A year after the Brookline Project began, a four-year Logo project for children ranging from three to nine years of age was begun at the Lamplighter School in Dallas, Texas. One of the objectives of this project was to determine if Logo could be used by students to learn better thinking or
problem solving. Students in the Lamplighter School third grade were randomly assigned to one of three homerooms, each having two computers with three more computers located in a shared space area. The students in two of the classes received one half hour of training in Logo per week and the other class received one hour of training in Logo per week. The study lasted approximately eight months. At the end of the study, fifteen students, five from each third grade homeroom, were randomly selected from the forty-nine students involved in the study. The experimenter gave these fifteen students a rule-learning task. The students from the group which had received one hour of training in Logo each week performed significantly better than the other two groups. This suggests that learning to program in Logo does improve problem-solving skills if enough time is spent on the computer (17).

Testing was also conducted in the New York City public schools as a result of a Logo program established by the New York Academy of Science. The project involved students in grades two through nine from a wide range of socioeconomic backgrounds. The staff at the schools involved in the project believe it has been a success in 90 to 95 per cent of the classrooms. Michael Tempel, project coordinator, stated, "The positive educational benefit was obvious! Kids were engaged in valid intellectual and social processes. You could see them developing..." (51, p. 130)
The final reports of the field tests of the above four Logo projects, based on the results of standardized tests, researcher developed measures, rule-learning tasks and/or observation, indicated that experience with Logo had a significant effect on learning. According to Lough,

In particular, Logo helped students to: (1) develop logical thinking and problem-solving skills; (2) learn to develop and test their own ideas and theories; and (3) become familiar with concepts such as variables, symmetry, angles and geometric forms (27, p. 50).

Summary

A 1983 survey of U.S. school districts revealed that the number of computers in use by public schools has more than doubled in the last year (29). According to another survey (49), there appears to be a definite trend toward using the computer to teach programming skills in the elementary school. At the same time, there is an increased emphasis on reasoning and problem solving in the schools (8, 22). As the initial research suggests, instruction in computer programming may enhance problem-solving abilities. However, further research in this area is required before any conclusive evidence can be gained.

The review of the literature pertinent to this study failed to reveal which language, if any, is most appropriate for developing problem-solving skills. The published re-
development of problem-solving skills revealed contradictory evidence. Those studies that did produce similar results tended to involve small sample sizes, and/or private school populations and/or no control group. Similarly, the published studies on BASIC and problem solving involved small sample sizes and/or no control group. Recommendations gleaned from existing research suggest a need for further and more refined research in this area.
CHAPTER BIBLIOGRAPHY


CHAPTER III

METHODS AND PROCEDURES

The study was conducted in several phases. First, objectives for the study were written and appropriate instructional materials were selected to teach these objectives. Second, a test to measure the general problem-solving skills of fifth graders was selected. Third, subjects were obtained and grouped for the BASIC, Logo and no programming treatments. Next, teachers were trained to conduct the activities. A computer-usage questionnaire and pretest were administered. The treatment sessions were held and a posttest was given. Finally, the data were analyzed to test the hypotheses of the study.

Objectives and Materials Selection

A list of BASIC and Logo system commands and program statements or primitives appropriate for use with fifth grade children was derived after a survey of several computer programming texts (3, 8, 10, 14) and articles (5, 6, 7, 9, 12, 15). These items were reviewed by three experts in the field—all computer educators. At least two experts had to agree upon an item for it to be included in the study. The broad objective of the study was then written as the performance mastery of each item by members of the programming groups (see Appendix A).
In order to accomplish the above objective, two publications were selected whose content most closely matched the list of programming objectives selected as appropriate for teaching fifth grade students. The approaches were similar to each other in that they guided students through problem situations and then gave a new problem for them to solve using what they had previously learned. It was necessary for the researcher to modify each publication to ensure that all objectives were covered. Modifications to the BASIC text were reviewed and approved by the author of the text. The modifications to the Logo text were adapted from *Forty Easy Steps to Programming in BASIC and Logo* (10) and *Logo for the Apple II* (2). All materials were tested for clarity in a workshop prior to the study.

The *Computer Challenge Kit* (3) was selected for use with the group of children learning BASIC. The modifications made by the researcher may be found in Appendix B. *Logo in the Classroom* (8) was selected for use with the group of children learning Logo. The modifications made by the researcher may be found in Appendix C.

**Instrument**

The instrument selected for use in this study as the pretest and posttest measure was the *New Jersey Test of Reasoning Skills* (NJTRS) (11). At the time of this study variant forms of the test were not available; so the same
test was given as the pre- and posttest measure. Since over nine weeks separated each administration of the test, practice effects should not be a factor.

The NJTRS is a 50-item test of elementary reasoning and inquiry skills, published in 1983 after seven years of extensive experimental research. It includes multiple choice questions that cover 23 areas of reasoning skills such as detecting assumptions, induction and recognizing symmetrical relationships. The test was written at the fourth-to-fifth grade reading level. The Flesch reading level is 4.5 and the Fogg is 5.0. According to data collected by Montclair State College, the average raw score for a fifth grade student is approximately 35.

The content validity of a test depends upon the degree to which it is representative of the domain it purports to test. For a test of reasoning skills to have content validity, there would have to be a taxonomy of the domain of reasoning skills. An attempt was made by Shipman (11) to construct a taxonomy in terms of the skills needed to perform the operations in the discipline of logic as these skills relate to linguistic usage. The taxonomy appears to be reasonably representative of the domain and the items selected for the New Jersey test appear to be reasonably representative of the taxonomy.
The reliability of the New Jersey Test compares favorably with established tests. Reliability ranges from .84 and above in grade five to .91 and above in grade seven.

Selection of the Sample

Approval to conduct the study was obtained from the central administrators of a wealthy ($18,500—approximate beginning teacher salary in 1983) suburban school district. The researcher selected two schools for inclusion in this study. Both schools had student populations that were approximately 90% white. The math computation and reading comprehension scores on the last norm-referenced test taken by students in both schools were skewed toward the top and middle of the national quartile. Also, the scores of the students at each school on the last state criterion-referenced exam were similar in the areas of math and reading. The principals of these schools were consulted and both agreed to allow their teachers and students to participate in the study.

Time

Permission was given to conduct the study in the spring for a period of seven weeks. The study began on March 26, 1984, and was concluded on May 18, 1984. The programming students received approximately 37 hours of computer instruction.
Schools

One school was selected as the control school because its students did not have access to district-owned computers at the time of the study. The other school was selected as the experimental school because it had recently received but not used computers for instructional purposes. Both schools had open area classrooms with similar facilities and curriculum. The children came from mainly blue-collar families.

Classrooms

All fifth grade teachers in both schools agreed to participate in the study. Thus, a total of nine fifth grade classrooms was involved. Four of the classrooms were experimental and five were control.

Population

The subjects of the study consisted of all regular fifth grade students in the four classrooms designated as being experimental and the five designated as being control. Because of scheduling problems with special classes, no students categorized as learning disabled (students pulled out for all or large portions of the school day to attend special classes) were involved in the study.
Procedures for Grouping Students

Five of the fifth grade classrooms, the entire grade level of one school excluding the learning disabled, served as the control group. This group received no in-school computer instruction at all for the duration of the study.

Four of the fifth grade classrooms, the entire grade level of the second school excluding the learning disabled, served as the experimental group. These classes were taken as intact classrooms and randomly assigned to the two treatments. These intact classrooms were used in their homeroom groupings so that all different academic ability levels were fairly evenly distributed throughout each classroom. Two slips of paper, one containing the word Logo and the other containing the word BASIC, were placed in a cup labeled "programming" and shuffled. One slip of paper was drawn from the programming cup. Next, four slips of paper, each containing the name of a fifth grade teacher, were placed in a cup labeled "teachers" and shuffled. Two slips of paper were drawn from the teacher cup. These two teachers' classrooms were assigned to the programming group written on the first piece of paper drawn from the programming cup. The other two classrooms were then assigned to the other programming group.

A total of 194 fifth grade students was initially involved in the study. However, 62 students were excluded from the final data analysis for various reasons. To be included in the study, the student must not have previously had
programming experience. The computer-usage questionnaire (see Appendix F) administered at the beginning of the study revealed that 44 students had previously had one or more of the following experiences: (a) private tutoring on the computer, (b) attendance at a computer camp, (c) programming experience in BASIC, Logo or some other language. Data on these students were removed for the final analysis. In addition, to be included in the study, a student had to be present for the pretest, at least 32 of the 37 treatment sessions, and the posttest. Though students were allowed to miss five days throughout the treatment sessions, if three consecutive days were missed, the students were dropped from the study. At the end of the treatment sessions, 18 more students were excluded from the final analysis for one or more of the above reasons. A total of 132 students qualified as subjects for the study. In the BASIC group, 36 students completed the programming sessions. In the Logo group, 31 students completed the programming sessions. The control group had 65 students that were included in the final data analysis.

**Procedures for Collection of Data**

Before the study began, training sessions were held for the participating teachers. Also, a computer-usage questionnaire was administered to the students prior to the study. The *New Jersey Test of Reasoning Skills* was administered as a
pretest and a posttest with a treatment period of seven weeks plus two weeks of school holiday time between each administration.

**Teacher Training**

Two fifth grade teachers volunteered to be the computer teachers for the study. The teacher who volunteered to teach the BASIC programming group had prior experience in the use of the BASIC language. Likewise, the teacher who volunteered to teach the Logo programming group had prior experience with the Logo language. Four weeks prior to the study, both teachers, plus two fifth grade back-up teachers and two aides with little or no programming experience, attended an eighteen hour training session with the researcher. The session consisted of familiarizing the teachers with the computer system used in the study, explaining and giving instructions on how to administer the questionnaire and testing instrument and working through the materials they would be using to instruct the students. Teachers of the students in the control group were given instructions on how and when to administer the questionnaire and instrument used in the study.

**Questionnaire**

A computer-usage questionnaire (see Appendix F), developed by the researcher, was administered two weeks prior to the study. The questionnaire's major purpose was to
determine if students in the study had prior experience with computers and to what extent. As previously stated, several students were excluded from the study based on their responses to the questions concerning previous programming experiences.

**New Jersey Test of Reasoning Skills**

The New Jersey Test of Reasoning Skills was administered one week prior to the study. The same instrument was given again as a posttest the week after the treatment was completed.

The teachers followed the written instructions for administering the test which were included with the instrument. After each administration of the test, the investigator collected all the test booklets and the completed and unused answer sheets.

**Procedures for Conducting Treatments**

Prior to the study, all students were made aware of the behavior rules to be followed while in the computer lab. Also, all students in the experimental group were familiarized with the computer system used in the study by taking the computer tutorial program developed by the manufacturers of the computer system (1). This tutorial familiarized students with the keyboard and other features of the system. It did not include instruction in programming.
Computer Lab Arrangement

The computer lab was divided into two sections with a large opaque curtain separating each section. There were ten Apple IIe computers in each section. BASIC and Logo sessions were conducted simultaneously on separate sides of the curtain.

Students were paired by the teachers to work on the computer. Two things were considered as this was arranged. First, students who had personality conflicts or other behavior problems when together were not placed on the same computer. Second, a very fast learner and a very slow learner were not grouped together to prevent the possibility of frustrating one or both students. In the Logo group, four computers had three children sharing it the majority of the time. The other six computers had two children. In the BASIC group, three computers had three children sharing it the majority of the time. The other seven computers had two children.

BASIC and Logo Sessions

For seven weeks treatment sessions were held for both the BASIC and Logo groups using the materials selected for the study. Each lesson/session had specific exercises that students were required to do to help them learn a particular skill. In addition to these exercises, the students were assigned two graphic projects (see Appendix D). Each group
received one hour of instruction or computer time each school day. A check sheet (see Appendix E) containing each student's name and the objectives for the study was marked by the teacher each time an objective was mastered. An objective was considered mastered when a student could use it in a programming situation without teacher assistance.

The researcher visited the computer teachers every two weeks. During these bi-weekly visits, feedback from the teachers was obtained to help assure that the procedures of the study were being followed.

Classroom Schedule Changes

In order to conduct the study during regular school hours, some changes were made in the amount of time the students in the experimental groups spent in regular classes over the seven-week period of the study. This was achieved by reducing the number of minutes per day spent in four subject areas. Math time was reduced 15 to 20 minutes a day (12.3 hours total), science and social studies time was reduced 5 minutes per day (6 hours total) and language arts time was reduced 30 minutes per day (18.5 hours total). The normal time allotment for math and language arts was one hour per day. The normal time allotment for science and social studies was 35 minutes per day. No changes were made in the control group's regular schedule.
Research Design

One of the most often used experimental designs in educational research involves an experimental group and a control group which are both given a pretest and posttest and in which the groups do not have pre-experimental sampling equivalence. This quasi-experimental design, the nonequivalent control group design described by Campbell and Stanley (4), was used in this study. It is especially suited to studies using naturally assembled collectives such as classrooms in which the groups are as similar as availability allows. In fact, the more similar the groups are in recruitment, the more effective will be the control. This design controls for the limitations of history, maturation, testing effects and instrumentation. Maturation and regression may pose internal validity problems. Also, possible interaction between selection and maturation may be an added limitation. Campbell and Stanley (4, p. 219) suggest the application of analysis of covariance to this design which necessitates the assumptions of homogeneity of regression and variance.

In educational research, the nonequivalent control group design is preferable to a true experimental design. In a true experimental design, students are randomly assigned and taken out of the natural classroom setting for treatment. This greatly increases the subjects' awareness of the experiment and threatens external validity. Campbell and Stanley posit that if the subjects consist of two natural groups and
there is no reason to suspect differential recruitment, even though the groups differ in initial means on control, the study may still approach true experimentation (4, p. 220).

Procedures for Analysis of Data

At the conclusion of the experiment, the pre- and posttests were returned to the IAPC Test Division at Montclair State College for scoring. When the printout of the scores from Montclair State College was received, scores from the pretest, which served as the covariate, and the posttest were analyzed using the SPSSX package (13) on analysis of covariance available at the Computing Center at North Texas State University.

The hypotheses of the study were restated in the null form. An alpha level of .05 was used to determine if an hypothesis was to be rejected.
CHAPTER BIBLIOGRAPHY


7. Doorly, A. **Microcomputers for gifted microtots.** G/C/T, 1980, 3(14), 62-64.


CHAPTER IV

RESULTS

Questionnaire Data

A computer-usage questionnaire (see Appendix F) was administered to 194 fifth grade students prior to the study. Students who had previously had private tutoring on the computer, had attended a computer camp, or knew BASIC, Logo or some other programming language were excluded from the study. Based on responses to the questionnaire, 44 students were excluded as a result of these criteria. Other factors such as missing too many programming sessions or missing pre-or posttest scores resulted in excluding 18 more students. Thus, the final total for data analysis was 132 subjects. Demographic data for these subjects are reported in Table I.

TABLE I

DEMOGRAPHIC DATA
(N=132)

<table>
<thead>
<tr>
<th>Group</th>
<th>Control (65)</th>
<th>BASIC (36)</th>
<th>Logo (31)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td>Male</td>
<td>49%</td>
<td>42%</td>
</tr>
<tr>
<td></td>
<td>Female</td>
<td>51%</td>
<td>58%</td>
</tr>
<tr>
<td>Average age</td>
<td>11.26</td>
<td>11.23</td>
<td>11.51</td>
</tr>
</tbody>
</table>
According to the data reported in Table I, all groups had more females than males. Also, the average age of the subjects was approximately 11 years.

Data were gathered to determine student access to computers prior to the study. As previously stated, several students were excluded from the final data analysis because of prior exposure to programming. The results of the remaining data pertaining to student access to computers are reported in Table II.

<table>
<thead>
<tr>
<th>TABLE II</th>
</tr>
</thead>
<tbody>
<tr>
<td>STUDENT ACCESS TO COMPUTERS</td>
</tr>
<tr>
<td>(N=132)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type of access</th>
<th>Per Cent</th>
<th>Per Cent</th>
<th>Per Cent</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Control</td>
<td>BASIC</td>
<td>Logo</td>
</tr>
<tr>
<td>Used computer</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>At friend's house</td>
<td>60</td>
<td>58</td>
<td>68</td>
</tr>
<tr>
<td>At store</td>
<td>40</td>
<td>44</td>
<td>45</td>
</tr>
<tr>
<td>Computers in home</td>
<td>43</td>
<td>39</td>
<td>32</td>
</tr>
<tr>
<td>Computers in home</td>
<td>13</td>
<td>8</td>
<td>19</td>
</tr>
<tr>
<td>Apple</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Atari</td>
<td>5</td>
<td>3</td>
<td>10</td>
</tr>
<tr>
<td>Commodore</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>IBM</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Radio Shack</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>TI</td>
<td>1</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>Other</td>
<td>3</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>How long?</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt; 1 year</td>
<td>5</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>1-2 years</td>
<td>6</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2-3 years</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>&gt; 3 years</td>
<td>3</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Hours used per day</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>&lt; 1 hour</td>
<td>6</td>
<td>8</td>
<td>13</td>
</tr>
<tr>
<td>1-2 hours</td>
<td>6</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>2-3 hours</td>
<td>0</td>
<td>0</td>
<td>3</td>
</tr>
<tr>
<td>&gt; 3 hours</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
According to the results of the questionnaire, over one half of the students included in the study had some exposure to the computer prior to the study. A few of the students in each group had access to one or more computers in their homes. The Atari Home Computer and the TI 99/4A were the most frequently mentioned computers in the students' homes. Most computers had been owned for less than one year and most students spent less than one hour per day on the computer.

Attitude data were also gathered from the questionnaire. Table III contains the results of the attitude questions.

<table>
<thead>
<tr>
<th>Attitude toward computer</th>
<th>Per Cent Control</th>
<th>Per Cent BASIC</th>
<th>Per Cent Logo</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enjoy using</td>
<td>47</td>
<td>56</td>
<td>64</td>
</tr>
<tr>
<td>Want one at school</td>
<td>60</td>
<td>47</td>
<td>42</td>
</tr>
<tr>
<td>Enjoy computer games</td>
<td>46</td>
<td>58</td>
<td>61</td>
</tr>
<tr>
<td>Glad to get computers in school</td>
<td>88</td>
<td>42</td>
<td>55</td>
</tr>
<tr>
<td>Wish had more time on computer</td>
<td>49</td>
<td>92</td>
<td>87</td>
</tr>
<tr>
<td>Not interested in computers</td>
<td>9</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Percentages are rounded.
According to the data in the preceding table, most children expressed some enjoyment of or interest in computers prior to the study. Six students (9% in the control group) expressed no interest in computers.

Statistical Analysis

The purposes of this study were to determine if learning to program a computer in either BASIC or Logo improves problem-solving skills of fifth grade students, and to determine if learning to program a computer in Logo is more effective than learning to program in BASIC for improving problem-solving skills in fifth grade students. To carry out these purposes, the following hypotheses were tested.

1. Students in the BASIC programming group will achieve significantly higher posttest scores on the New Jersey Test of Reasoning Skills than will students in the control group.

2. Students in the Logo programming group will achieve significantly higher posttest scores on the New Jersey Test of Reasoning Skills than will students in the control group.

3. Students in the Logo programming group will achieve significantly higher posttest scores on the New Jersey Test of Reasoning Skills than will students in the BASIC programming group.

The independent variable was the treatment condition (BASIC or Logo). The dependent variable was the number of correct responses on the New Jersey Test of Reasoning Skills,
which was used as the posttest measure. Used as a pretest, this instrument also served as the covariate in the study. A check was made for homogeneity of variance and regression and both assumptions were met. All hypotheses were tested in the null form.

Data pertaining to all three hypotheses are presented in Table IV. These data indicate that a significant difference was not found at the .05 level. The null hypothesis was retained in all three cases. Learning to program in BASIC or Logo does not significantly improve problem-solving ability as measured by the New Jersey Test of Reasoning Skills when compared to a control group that receives no programming instruction.

<table>
<thead>
<tr>
<th>Source</th>
<th>Sum of Squares</th>
<th>df</th>
<th>Mean Squares</th>
<th>F</th>
<th>Level of Significance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main effects</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Group</td>
<td>32.84</td>
<td>2</td>
<td>16.42</td>
<td>0.70</td>
<td>0.50</td>
</tr>
<tr>
<td>Residual</td>
<td>3002.95</td>
<td>128</td>
<td>23.46</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>7380.81</td>
<td>131</td>
<td>56.34</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The adjusted posttest means were 33.50 for the control group, 32.92 for the BASIC group and 34.32 for the Logo
group. Though no significant differences were found among the posttest means of the three groups, a significant gain was found between the pretest and posttest means of each group. Data pertaining to this are reported in Table V.

### TABLE V

GAINS IN PROBLEM SOLVING FROM PRETEST TO POSTTEST

<table>
<thead>
<tr>
<th>Source</th>
<th>Pretest Means</th>
<th>Posttest Means</th>
<th>Standard Deviation</th>
<th>df</th>
<th>2-Tail Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control (65)</td>
<td>31.1538</td>
<td>33.7385</td>
<td>5.154</td>
<td>64</td>
<td>0.001</td>
</tr>
<tr>
<td>BASIC (36)</td>
<td>30.8056</td>
<td>32.8889</td>
<td>6.143</td>
<td>35</td>
<td>0.049</td>
</tr>
<tr>
<td>Logo (31)</td>
<td>30.2581</td>
<td>33.8710</td>
<td>3.593</td>
<td>30</td>
<td>0.001</td>
</tr>
</tbody>
</table>

Maximum score = 50.

As the data in Table V indicate, a statistically significant gain occurred from the pretest means to the posttest means of each group. An average gain of 2.58 for the control group between the pretest and posttest means was significant beyond the .001 level. An average gain of 2.08 for the BASIC group between the pretest and posttest means was significant beyond the .05 level. An average gain of 3.61 for the Logo group between the pretest and posttest means was significant beyond the .001 level. Problem-solving ability significantly improved from the pretest to the posttest for all three groups.
CHAPTER V

DISCUSSION

The problem of this study was to compare the effects of learning two programming languages, BASIC or Logo, on the problem-solving abilities of fifth grade students. The programming population consisted of students from one experimental school. The control school was comparable to the experimental school in facilities, curriculum and student population. The teachers of the experimental groups, using the materials provided (1, 11), conducted the programming sessions for a period of approximately seven weeks. The New Jersey Test of Reasoning Skills (8), which served as the pretest measure, was then administered as the posttest measure. Analysis of covariance was the statistical technique applied to the data.

The results of the study are discussed in terms of findings, interpretations and conclusions. Suggestions for further research conclude the chapter.

Findings

This study examined whether learning to program a computer in BASIC or Logo was more effective than no programming instruction in developing problem-solving skills.
It also examined whether learning to program the computer in one language was more effective than in the other for enhancing the development of these skills. The results indicated that learning to program the computer did not significantly increase problem-solving skills when compared to a control group which received no programming instruction. Also, one language did not appear to significantly enhance problem-solving skills more than the other. A report of these results is presented in Table IV of Chapter IV.

Even though no significant differences were found among the posttest scores of the three groups, it should be noted that significant gains were found from the pretest means to the posttest means of each group. A report of these results is presented in Table V of Chapter IV.

Interpretations

Much of the current research tends to support the contention that learning a programming language enhances the development of problem-solving skills. Rose (7) concluded that learning to program in BASIC develops problem-solving skills in fifth grade students in areas such as hypothesis testing and determination of relevance of data. Gorman (5) concluded that learning to program in Logo significantly improved the ability of third graders to perform a rule-learning task. Similarly, some schools in various places throughout the nation which are conducting Logo projects (12)
have concluded that learning to program in Logo develops problem-solving skills in children in grades two through nine. However, the present study does not support the above findings. The data collected for this study indicate that learning to program in BASIC or Logo does not significantly increase problem-solving skills when compared to a control group receiving no programming instruction as measured by the NJTRS.

Even though the study conducted by Rose involved predominately white, middle class fifth graders as did the present study, there are two factors that may explain the differences obtained in the results of the two studies. One difference involves the amount of regular instruction time missed in order to use the computer. The experimental group in Rose's study received their computer instruction before and after regular school hours. Therefore, the fifth graders in that study did not miss any regular classroom instruction as did the students in the present study. In order to conduct the current study during regular school hours, the amount of time spent in the regular classroom was reduced for the experimental groups for a period of approximately seven weeks. Science and social studies were reduced 5 minutes per day (6 hours total), math was reduced 20 minutes per day (12.3 hours total) and language arts was reduced 30 minutes per day (18.5 hours total). The control group received the regular amount of clock hours of classroom instruction in
science, social studies, math and language arts. Since the subjects in the Rose study missed no regular classroom instruction, this may be a possible explanation of the inconsistent results between the two studies. Also, the 37 extra clock hours in these subject areas may have accounted for the increase in the control group's posttest scores in the current study.

A second difference between the Rose study and the present study involves the instrument used. Rose used the Cornell Critical Thinking Test (4) and the Developing Cognitive Abilities Test (10) as a pretest and posttest measure, whereas the present study used the New Jersey Test of Reasoning Skills. It is possible that these instruments measure different problem solving skills. The NJTRS may not measure the type of problem solving developed when a programming language is learned whereas the tests used in Rose's study may measure these abilities. It is also possible that the problem-solving abilities that programming may develop are too fine or small to be detected by the instrument used in the present study.

Another point of concern about the instrument used in the current study deals with the average score of fifth graders as reported by Montclair State College. The reported average score for fifth graders is 36. However, the highest average score attained by students in the current study was 33.87. As stated previously, the students in the present
study had achievement test scores that were skewed toward the top and middle of the national quartile. From these scores, it could be assumed that the students in the current study would perform at the average on the NJTRS. However, the average scores for all three groups fell below the reported average score for fifth graders. Information was given that approximately 1,200 students were involved in the attainment of the average score on the NJTRS. However, no information was given as to where these scores were obtained. It appears that researchers should look with scrutiny upon the average score for the NJTRS as reported by Montclair State College.

Gorman (5) conducted a study in a private school in which he assessed the effects of learning to program in Logo on the problem-solving abilities of third graders. He utilized a rule-learning task to determine the effects. Gorman found that learning to program in Logo did significantly increase problem-solving ability if enough time were spent in Logo instruction. The students who showed a significant gain spent one hour a week for nine months in Logo instruction. Instructional time was missed during homeroom classes only.

Several differences between the Gorman study and the present study may account for the different results obtained. First, the Gorman study used a different means of measuring problem solving which may be more appropriate for detecting improvement in problem-solving skills than the NJTRS. Second, unlike the present study, no math or language arts
instructional time was missed in order to conduct the study. Third, the Gorman study was conducted using third grade students whereas the present study used fifth graders. This age difference may be a possible reason for the inconsistent results of the two studies. The average age of all subjects in the current study was 11.31 years. At about the age of 11, according to Piaget (2, 3), some children progress into the formal operations stage of cognitive development. During the period of formal operations, children acquire the ability to make complex deductions, analyze ways of reasoning and solve problems by systematically testing hypothetical situations. Children break away from the need for concrete representations gathered by their senses and become able to use their thoughts to hypothesize beyond the real and reason scientifically. Since all subjects were approximately the same age, 11, in the present study, and the students in the Gorman study were approximately 8 years old, this may account for the Gorman study finding significant differences and the current one not. The maturation aspect may have clouded the results of the present study. This phenomenon could also explain why all three groups had posttest means that were not significantly different though significant gains were made from the pretest to the posttest scores.

Research conducted by Statz (9) in the early 70's produced results similar to the current study. She found that teaching nine to 11 year olds how to program in Logo did not
significantly increase the development of problem-solving skills. Though some of the students in her study were slightly younger than the students in the present study, the maturation aspect may have affected her results, too.

Even though several possible explanations for the results of the current study have been discussed, the results obtained in this study indicate that learning to program in BASIC or Logo does not significantly improve problem-solving ability as measured by the New Jersey Test of Reasoning Skills when compared to a group that receives no programming instruction.

Conclusions

While the programming treatment in this study did not appear to promote significant differences among the groups in the development of problem solving as measured by the instrument used, it does not mean that learning to program is not an effective way of enhancing the development of these skills. It should be noted that the students in the experimental groups improved significantly from the pretest to the posttest in problem solving even though they missed over 37 hours of regular classroom instruction. It can be implied that missing this amount of regular instruction does not significantly detract from the students' ability to problem solve as measured by the New Jersey Test of Reasoning Skills.
Since no significant difference was found between the posttest means of the programming groups, it may be concluded that Logo is not significantly more effective than BASIC when working with fifth grade students for enhancing the development of problem solving as measured by the NJTRS. This coupled with the fact that both programming groups improved significantly from pretest to posttest implies that one language is not more appropriate for use with fifth graders. Holzman and Glaser (6) reached a similar conclusion with a study involving sixth graders.

Suggestions for Further Research

In light of the results of this study, the following possibilities for further research are suggested.

1. This study examined the effects of learning a programming language on problem-solving skills with fifth grade students only. Therefore, further studies should be conducted with younger children to determine whether the effects vary with different stages of cognitive development.

2. Even though the experimental groups missed over 37 hours of classroom instruction, it did not significantly affect their ability to problem solve. It would be desirable to determine if their achievement in the areas in which classroom instruction was reduced (science, social studies, math and language arts) was affected.
3. Since there is no conclusive evidence on what type of problem-solving skills learning how to program may develop, it is suggested that measures other than or in addition to the test utilized in this study be explored in future studies.

4. This study was conducted in a mostly white, affluent school district. Different results may be obtained if the study is conducted in a more integrated, less affluent area.

5. This study involved an intensive seven-week programming session. It is possible that different results might be found if the treatment were spread over the nine-month school year.


APPENDIXES
APPENDIX A

STUDY OBJECTIVES
**STUDY OBJECTIVES**

The students in the experimental group will be able to use the following system commands and program statements at the end of the study:

<table>
<thead>
<tr>
<th>BASIC group</th>
<th>LOGO group</th>
</tr>
</thead>
<tbody>
<tr>
<td>CATALOG</td>
<td>CATALOG</td>
</tr>
<tr>
<td>LOAD</td>
<td>LOAD</td>
</tr>
<tr>
<td>SAVE</td>
<td>SAVE</td>
</tr>
<tr>
<td>HOME</td>
<td>HOME</td>
</tr>
<tr>
<td>LIST</td>
<td>POPS</td>
</tr>
<tr>
<td>NEW</td>
<td>POTS</td>
</tr>
<tr>
<td>PRINT</td>
<td>ERASE</td>
</tr>
<tr>
<td>INPUT</td>
<td>PRINT</td>
</tr>
<tr>
<td>IF/THEN</td>
<td>READLIST</td>
</tr>
<tr>
<td>FOR/NEXT/STEP</td>
<td>IF</td>
</tr>
<tr>
<td>END</td>
<td>REPEAT</td>
</tr>
<tr>
<td>GOTO</td>
<td>END</td>
</tr>
<tr>
<td>TEXT</td>
<td>FULLSCREEN</td>
</tr>
<tr>
<td></td>
<td>SPLITSCREEN</td>
</tr>
<tr>
<td></td>
<td>TO</td>
</tr>
<tr>
<td></td>
<td>EDIT</td>
</tr>
<tr>
<td></td>
<td>MAKE</td>
</tr>
<tr>
<td></td>
<td>WORD</td>
</tr>
<tr>
<td></td>
<td>SENTENCE</td>
</tr>
</tbody>
</table>

**APPLESOFT GRAPHICS**

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>GR</td>
</tr>
<tr>
<td>HGR</td>
</tr>
<tr>
<td>COLOR</td>
</tr>
<tr>
<td>HCOLOR</td>
</tr>
<tr>
<td>PLOT</td>
</tr>
<tr>
<td>HLIN</td>
</tr>
<tr>
<td>VLIN</td>
</tr>
</tbody>
</table>

**TURTLE GRAPHICS**

<p>| |</p>
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CS</td>
</tr>
<tr>
<td>SETPC</td>
</tr>
<tr>
<td>SETBG</td>
</tr>
<tr>
<td>FD</td>
</tr>
<tr>
<td>BK</td>
</tr>
<tr>
<td>RT</td>
</tr>
<tr>
<td>LT</td>
</tr>
<tr>
<td>SETH</td>
</tr>
<tr>
<td>PU</td>
</tr>
<tr>
<td>PD</td>
</tr>
<tr>
<td>PE</td>
</tr>
<tr>
<td>HT</td>
</tr>
<tr>
<td>ST</td>
</tr>
<tr>
<td>SETX</td>
</tr>
<tr>
<td>SETY</td>
</tr>
<tr>
<td>SETPOS</td>
</tr>
</tbody>
</table>
APPENDIX B

BASIC MODIFICATIONS
A FOR/NEXT loop tells the computer to do one thing many times. It is like getting a computer to go around in a circle. To see how this works, insert the Challenge Disk and

Type:

RUN LOOP (RETURN)

Your screen should look like this:

1 2 3 4 5 6 7 8 9 10

Type:

LIST (RETURN)

Your screen should look like this:

5 HOME
10 FOR X = 1 TO 10
15 PRINT X
20 NEXT X
Line 5 clears the screen.
Line 10 tells the computer it will count from 1 to 10. The first number tells the computer where to start counting and the second number tells the computer where to stop. Line 15 tells the computer to print the value of X. The first time through the loop X will have a value of 1, the second time through the loop X will have a value of 2 and so on until the computer counts to 10. Line 20 tells the computer to go to line 10, add 1 to X and go through the program again. This line also tells the computer to stop when it reaches 10.

Now change line 10.

Type:
10 FOR X = 10 TO 20 (RETURN)

RUN (RETURN)

Beside number 1 on your worksheet write with what number the computer began counting and with what number it stopped counting.

The computer can count by two; 0, 2, 4, 6, 8, 10 and so on. It can also count by five; 0, 5, 10, 15, 20 and so on. It can count by any number. STEP is used to do this.

Type:
10 FOR X = 1 TO 10 STEP 2 (RETURN)

RUN (RETURN)

Beside number 2 on your worksheet write what the program does.

Try this. Change line 10 to make the computer count from 0 to 100 by 5's.
Beside number 3 on your worksheet write the command you used to make the computer count from 0 to 100 by 5's.

In addition to counting up, the computer can count down or backwards. Try this.

Type:
10 FOR X = 10 TO 1 STEP -1 (RETURN)
RUN (RETURN)

Your screen should look like this:

10
9
8
7
6
5
4
3
2
1

Change line 10 to make the computer count backwards from 50 to 0 by 5's.

Beside number 4 on your worksheet write the command you used to make the computer count backwards from 50 to 0 by 5's.

Go to Challenge 7...
Red Unit
Lesson 9 Worksheet

1. With what number did the computer begin counting? _____
   With what number did the computer stop counting? _____

2. What did the program do when you added STEP 2 to it?
   ______________________________________________________

3. What command did you use to make the computer count from 0 to 100 by 5's?
   ______________________________________________________

4. What command did you use to make the computer count backwards from 50 to 0 by 5's?
   ______________________________________________________
To do this task you must know how to use GR, COLOR, HLIN, VLIN, and FOR/NEXT/STEP.

A. Insert the Challenge Disk and
Type:
RUN PAINT
Type:
TEXT (RETURN)
HOME (RETURN)
LIST (RETURN)

Your screen should look like this:

5 REM PROGRAM NAME IS PAINT
10 GR
15 COLOR = 15
20 FOR D = 0 TO 39
30 HLIN 0,39 AT D
40 NEXT D

Line 5 tells the program's name.
Line 10 puts the computer in graphics mode.
Line 15 sets the color to white.
Line 20 begins a loop that will start counting at 0 and stop at 39.
Line 30 tells the computer to draw a horizontal line from 0 to 39 at D. The first time through D will be 0, the second time through D will be 1 and so on until D is 39.
Line 40 tells the computer to go to line 20, add 1 to D and
go through the program again. This line also tells the computer to stop looping when D reaches 39.

B. Use the STEP command to change the program so that it only paints every other line. The screen should look like it has stripes on it.

C. Write your own program that paints white vertical stripes and blue horizontal stripes on the screen. Use FOR/NEXT/STEP to do this. Type STRIPES to see a program like this.
APPENDIX C

LOGO MODIFICATIONS
SESSION 21
NUMBERS, WORDS, AND LISTS

PART I--NUMBERS

You have already seen examples of using numbers in turtle programs. Logo provides the basic arithmetic operations:

addition +
subtraction -
multiplication *
division /

When you combine arithmetic operations, multiplications and divisions are performed before additions and subtractions unless you use parentheses. Here are some examples:

Type:

PRINT 3 + 2 * 5 (RETURN)
13

Notice that the computer multiplied 2 times 5 and then added 3.
Try this.

Type:

PRINT (3 + 2) * 5 (RETURN)
25

Notice that the computer added 3 plus 2 first (because it is in parentheses) and then multiplied the result by 5.

ASSIGNMENT: Do Problem Sheet for SESSION 21/PART I.
PART II--WORDS and LISTS

In Logo, strings of characters are called words. To indicate a word in Logo, you type the character string with a quotation mark in front of it. For example,

Type:

PRINT "HELLO" (RETURN)
HELLO

Notice that the quotation mark goes only at the beginning of the word.

Logo has a WORD operation. It takes two words and combines them to form a single word.

Type:

PRINT WORD "SUB" "MARINE" (RETURN)
SUBMARINE

In Logo, a sequence of words is called a list. You must separate words in a list by at least one space and the entire list must be enclosed in brackets. For example,

Type:

PRINT [THIS IS A LIST] (RETURN)
THIS IS A LIST

Notice that you do not use quote marks when you use brackets and the brackets are not printed when the list is.

Numbers can also be a list. For example,

Type:

PRINT [3 * 3] (RETURN)
3 * 3

Notice that the computer did not print the result of the calculation 3 times 3 but printed exactly what you typed inside the brackets.
SENTENCE is an operation that puts lists together. For example,

Type:

PRINT SENTENCE [THIS IS HOW] [A SENTENCE CAN BE MADE] (RETURN)
THIS IS HOW A SENTENCE CAN BE MADE

Type:

PRINT SENTENCE "THIS [WORKS TOO] (RETURN)
THIS WORKS TOO

Type:

PRINT SENTENCE "THIS "TOO (RETURN)
THIS TOO

ASSIGNMENT: Do Problem Sheet for SESSION 21/PART II.
Write the PRINT statement you used to get the answer beside each number. Use only the numbers 5, 6 and 10. For example, how will you use the numbers 5, 6 and 10 to get 22?

Answer: PRINT 5 + 6 + 10

1. 40

Answer:

2. 9

Answer:

3. 65

Answer:

4. 12

Answer:

5. 56

Answer:
1. How can you make the computer print HELLO followed by your name?

2. Use the WORD operation to make the computer print SIX and TEEN as one word.

3. Use the SENTENCE operation to make the following lists into a sentence:

[I LIKE]
[APPLE LOGO]
SESSION 22

READLIST — INPUT FROM THE KEYBOARD

It is possible to talk to the computer while a procedure is running. This is done by using the command READLIST. READLIST waits for you to type in a line and then prints the line you typed in as a list. For example,

Type:

TO TALK
PRINT [WHAT IS YOUR NAME?]
PRINT SENTENCE [HELLO,] READLIST
END

Press CONTROL-C and then type TALK.

The computer prints the message WHAT IS YOUR NAME? and then waits for you to type something. Type in your name. The computer then takes your name and prints HELLO, "your name."

Type TALK again but this time put in someone else's name. What happens? The computer does not know who you are. It just uses the information you give it to do what you program it to do.

You can use READLIST more than one time in a program. For example,

Type:

TO CHAT
PRINT [WHAT IS YOUR NAME?]
PRINT SENTENCE [HELLO,] READLIST
PRINT [HOW OLD ARE YOU?]
PRINT SENTENCE [YOU LOOK MUCH OLDER THAN] READLIST
END

Press CONTROL-C and type CHAT to see what this procedure does.

ASSIGNMENT: Do Problem Sheet for SESSION 22.
Write a procedure using READLIST that asks a person's name, address, and telephone number and prints messages with each.
SESSION 23

ASSIGNMENTS (MAKE) AND CONDITIONAL STATEMENTS (IF)

The Logo command used to name things is MAKE.

Type:

MAKE "NUMBER 1 (RETURN)
PRINT :NUMBER (RETURN)
1

In the first line you tell Logo that you are going to call the number 1 by the name NUMBER. The first input to MAKE is the name and the second input is the thing you are naming. If you wish to print the thing associated with the name, you must type a : (colon) and the name.

Try this.

Type:

MAKE "NUMBER 10 (RETURN)
PRINT :NUMBER (RETURN)
10

Type:

MAKE "COLOR "GREEN (RETURN)
PRINT :COLOR (RETURN)
GREEN

Type:

MAKE "COMPUTER "APPLE (RETURN)
PRINT :COMPUTER (RETURN)
APPLE

The Logo statement used to test for a condition is IF. Let's review recursion and variables before we use this statement.

Type:

TO COUNTUP :START (RETURN)
PRINT :START (RETURN)
COUNTUP :START + 1 (RETURN)
END (RETURN)

Press CONTROL-C and type COUNTUP 1.
To stop this procedure you must press CONTROL-G.

The first line tells the name of the procedure and indicates that we need to give START some value.

The second line tells Logo to print the value of START.

The third line tells Logo to call the procedure named COUNTUP and add 1 to the value of START.

The only way to stop a procedure written like this is by pressing CONTROL-G. Let's use the IF statement to improve this procedure.

Type:

EDIT "COUNTUP (RETURN)

TO COUNTUP :START
PRINT :START
IF :START = 100 [STOP] <-- add this line
COUNTUP :START + 1
END

Press CONTROL-C and type COUNTUP 50.

Logo will count from 50 to 100 and stop because of the line we added. The IF statement we added means if START equals 100 then stop. Of course if START doesn't equal 100 it will continue to execute. This could be a problem if you typed in COUNTUP 101. START would never have a value of 100 and you would have to stop the procedure with CONTROL-G.

Let's use the MAKE command and add two more lines to our procedure.

Type:

EDIT "COUNTUP (RETURN)

TO COUNTUP :START
MAKE "NUMBER :START <-- add this line
IF :NUMBER > 100 [STOP] <-- add this line
PRINT :START
IF :START = 100 [STOP]
COUNTUP :START + 1
END

Press CONTROL-C and type COUNTUP 101.
Nothing happens. This is because we made NUMBER have the value of START (101) and used the IF statement to test to see if NUMBER was greater than 100. In this case it was so the procedure stopped.

There is a way to do this procedure without using the MAKE command and using only one IF statement. Can you figure it out?

An IF statement can be used with strings. For example,

Type:

```
TO BRAG
PRINT [WHO'S THE SMARTEST PERSON IN THIS CLASS?]
IF READLIST = [ME] [PRINT [OF COURSE!] STOP]
PRINT [NO, TRY AGAIN]
BRAG
END
```

Press CONTROL-C and then type BRAG.

Logo will print WHO'S THE SMARTEST PERSON IN THIS CLASS? on the screen and wait for you to type in something. What command asks Logo to stop and wait for you to input something from the keyboard (READLIST)? If you type ME, Logo will reply OF COURSE! and the procedure will stop. If you type anything else other than ME, Logo will print NO, TRY AGAIN.

ASSIGNMENT: Do Problem Sheet for SESSION 23.
1. Write a procedure that will count backwards from some number input by the user.

2. Modify the above procedure using the IF statement to make it stop counting backwards when it reaches 0.

3. Write a procedure that asks your favorite color. Accept input from the keyboard and use an IF statement to test to see if your favorite color was input. If your favorite color was input, have the computer print a message saying that it is your favorite color too. When any other color is input, have the computer print a message saying I like that color but it is not my favorite. Use recursion. This procedure should not stop until your favorite color is input from the keyboard.
GRAPHIC PROJECTS

Each lesson/session has specific exercises that students are required to do to help learn the particular skills taught in each lesson. In addition to these exercises, the students will be assigned two graphic projects.

PROJECT I: Write a program/procedure to draw your initials on the screen. (BASIC group will do this in low resolution graphics.) The BASIC group should do this project after Lesson 7. The Logo group should do this project after Session 15.

PROJECT II: Write a program or set of procedures to make a detailed graphic display of your choice. (BASIC group will do this in high resolution graphics.) This project is to be done as the last activity of the study.
<table>
<thead>
<tr>
<th>NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>15</td>
</tr>
<tr>
<td>16</td>
</tr>
<tr>
<td>17</td>
</tr>
<tr>
<td>18</td>
</tr>
<tr>
<td>19</td>
</tr>
<tr>
<td>20</td>
</tr>
<tr>
<td>21</td>
</tr>
<tr>
<td>22</td>
</tr>
<tr>
<td>23</td>
</tr>
<tr>
<td>24</td>
</tr>
<tr>
<td>25</td>
</tr>
</tbody>
</table>
APPENDIX F

COMPUTER-USAGE

QUESTIONNAIRE
COMPUTER-USAGE QUESTIONNAIRE

NAME ____________________________________________________________

AGE ________

BOY ___ GIRL ___

1. Have you ever used a computer?
   ___ yes
   ___ no

IF YOU ANSWERED YES, PLEASE ANSWER QUESTION NUMBERS 2 AND 3.
IF YOU ANSWERED NO, PLEASE SKIP TO QUESTION NUMBER 7.

2. Please check each of the following statements that are true of you.
   ___ a. I have access to a computer in my home.
   ___ b. I have used a computer at a friend's house.
   ___ c. I have used a computer at a store.
   ___ d. I have had private tutoring on a computer.
   ___ e. I have attended a computer camp.
   ___ f. I enjoy using a computer.
   ___ g. I can write some programs in BASIC.
   ___ h. I can write some programs in Logo.
   ___ i. I can write programs in some language other than BASIC or Logo.
   ___ j. I wish we had computers at school.
   ___ k. I enjoy playing games on a computer.

3. Is there a computer in your home?
   ___ yes
   ___ no

IF YOU ANSWERED YES, PLEASE ANSWER QUESTIONS 4, 5, 6 AND 7.
IF YOU ANSWERED NO, PLEASE SKIP TO QUESTION NUMBER 7.

4. What kind of computer do you have at home?
   ___ a. Apple
   ___ b. Atari
   ___ c. Commodore
   ___ d. IBM
   ___ e. Radio Shack
   ___ f. TI
   ___ g. Other ____________________
5. How long have you had a computer?
   a. less than a year
   b. between one and two years
   c. between two and three years
   d. more than three years

6. How many hours a day do you use the computer?
   a. less than an hour
   b. between one and two hours
   c. between two and three hours
   d. more than three hours

7. Check the statements that are true of you.
   a. I will be glad when we get computers in our school.
   b. I wish I had more time to do things on the computer.
   c. I am not interested in computers.
BIBLIOGRAPHY

Books


Whimbey, A. Intelligence can be taught. New York: Dutton, 1975.


Articles


Doorly, A. Microcomputers for gifted microtots. G/C/T, 1980, 3 (14), 62-64.


Watt, D. Logo in the schools. Byte, 1982, 7 (8), 116-134.


Publications of Learned Organizations


Reports


Unpublished Materials
