A QUALITATIVE ANALYSIS OF THE COMPUTER PROGRAMMING
ABILITIES AND THOUGHT PROCESSES OF
FIVE-YEAR-OLD CHILDREN

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

Sandra N. Hines, B.A., M.A.
Denton, Texas
August, 1984

The problem of this study was to describe and analyze the computer programming abilities and thought processes of five-year-old children using a conventional microcomputer and the Apple LOGO language.

This dissertation reports on the behavior of five kindergarten children and the comments they made as they learned to program in LOGO on an Apple II Plus microcomputer. The five participants were randomly selected from a group of ten five-year-olds who passed a screening test of numeral and capital letter recognition. The sample included three girls and two boys, all of whom were white. The students met individually with the researcher and the computer for about twenty minutes every day during a ten-week period.

Each session was tape recorded, transcribed, and analyzed. The data were analyzed into five major categories of behavior with the computer: (1) conceptual difficulties, (2) perceptual difficulties, (3) physical difficulties, (4) affective responses, and (5) thought processes. Several instruments were also utilized to assess the developmental progress, their attitudes toward the computer, and the
programming competence demonstrated by the students during the study. They included a test of number quantity, nine Piagetian tasks, a student interview, two types of programming tests, and a parent questionnaire. Students' computer programs were printed out and analyzed in individual case studies.

The students' performance on the Piagetian tasks was compared to their performance on the LOGO programming tests and to the number of procedures they programmed during the study. There was no clear relationship between programming competence and developmental maturity.

The participants' computer programming abilities varied. All of them worked out procedures on the computer with the assistance of the researcher. The programming tests indicated that two students were independent programmers, two students could write procedures, but they had difficulty correcting their errors, and one student could not be considered a programmer. He preferred playing the preskill games on the computer.
# TABLE OF CONTENTS

**LIST OF TABLES** ........................................... vii

**Chapter**

I. INTRODUCTION .......................................... 1

  - Background of the Problem
  - Statement of the Problem
  - Purpose of the Study
  - Research Questions
  - Significance of the Study
  - Limitations of the Study
  - Definition of Terms
  - Procedures of the Study
  - Summary

II. REVIEW OF THE LITERATURE ............................. 14

  - The Computer Programming Abilities of Children
    - Summary
  - The Effects of Programming on Children
    - Summary
  - The Thought Processes of Young Children
    - The Piagetian View of Cognitive Functioning
    - Characteristics of Preoperational Thought
    - Conceptualization and Cognizance
    - Success and Understanding
    - Problem-solving Behaviors of Preoperational Children
    - Summary

III. METHODOLOGY ......................................... 39

  - The Pilot Study
    - Sample and Setting
    - Instrumentation
    - Teaching Procedures
    - Data Collection Procedures
    - Information from the Pilot Study
    - Conclusions of the Pilot Study
The Major Study
Sample and Setting
Instrumentation
Teaching Procedures
Analysis of the Data
Summary

IV. ANALYSIS OF DATA AND FINDINGS

Instruments
Screening Test
Piagetian Tasks
Student Interview
Parent Questionnaire
Programming Tests

Discussion of the Results of the Instruments

Field Notes and Transcriptions
Conceptual Difficulties
Conceptual Strengths
Perceptual Difficulties
Physical Difficulties
Affective Responses
Thought Processes

Discussion of the Analysis of the Field Notes

Case Studies of Student Programming
Alexandra
Andrea
Billy
Danielle
Jonathan

Discussion of the Case Studies
Computer Programs
Programming Style
Attitude Toward Programming

Discussion of the Instruments, Field Notes, and Case Studies

V. SUMMARY, FINDINGS, IMPLICATIONS, AND RECOMMENDATIONS

Summary
Findings
Implications
Recommendations for Further Research
APPENDICES

A Letter to Parents
B Screening Test
C Student Interview
D Parent Questionnaire
E Programming Tests
F Programming Commands for Dot Tasks
G Correct Resolution of Dot Tasks
H Analysis of Program Names
I Students' Questions About the Computer
J Alexandra's Programming
K Andrea's Programming
L Billy's Programming
M Danielle's Programming
N Jonathan's Programming

BIBLIOGRAPHY
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Sex, Birthdate, Age, and Absence of the Subjects</td>
<td>48</td>
</tr>
<tr>
<td>II. Comparison of Performance on the Piagetian Tasks</td>
<td>79</td>
</tr>
<tr>
<td>III. Pretest and Posttest Responses Classified According to Stages</td>
<td>80</td>
</tr>
<tr>
<td>IV. Comparison of Performances on the Programming Tests</td>
<td>101</td>
</tr>
<tr>
<td>V. Visual Discrimination Problems with Letters and Numerals</td>
<td>111</td>
</tr>
<tr>
<td>VI. Common Reversals</td>
<td>112</td>
</tr>
<tr>
<td>VII. Auditory Discrimination Problems with Naming Numerals</td>
<td>115</td>
</tr>
<tr>
<td>VIII. Comparison of Individual Attention Spans During Programming</td>
<td>134</td>
</tr>
<tr>
<td>IX. Numbers Most Frequently Used in Programming</td>
<td>141</td>
</tr>
<tr>
<td>X. Comparison of Program Names for the House Task</td>
<td>149</td>
</tr>
<tr>
<td>XI. Comparison of Shape Procedure Names</td>
<td>150</td>
</tr>
<tr>
<td>XII. Analysis of Program Names</td>
<td>235</td>
</tr>
</tbody>
</table>
CHAPTER I

INTRODUCTION

Background of the Problem

The use of microcomputers has become commonplace at many grade levels in schools across the United States during the past few years. According to a recent survey there are 325,000 microcomputers in public schools. Of that number 110,000 are in use in elementary schools. A study conducted by Market Data Retrieval (1983) reported that 55,765 public schools use computers for instruction. This is more than twice the number that used microcomputers for instruction in the previous year. More computers continue to be purchased for use in the schools. As advancements in technology and increased competition have reduced the price of microcomputers, more schools are allocating microcomputers to the primary grades and to the kindergarten.

School administrators and teachers are confronted with a host of concerns related to the introduction of this new technology in the classroom. There is a fundamental question of how the computer is to be used by students—for computer-assisted instruction, for programming, or both? There is also concern about introducing yet another subject into the "basics" which the schools must teach. How can
instruction in the use of computers be justified aside from its popularity? What are the cognitive benefits and the affective consequences on students who use microcomputers at school? Educators ponder which grade level should receive the initial instruction in computer literacy and what skills this specialized curriculum should include. These important curricular questions deserve answers that perhaps can be provided by research. This study originated from a shared concern about these issues.

Computers in schools are being used in a variety of ways. They perform many useful managerial tasks. School administrators use them to keep attendance records, balance the budget, and assist in class scheduling. Teachers may use microcomputers as a grade book to keep records of student performance. They may also use the computer as a diagnostic device by entering a description of student performance and obtaining an instructional prescription for each student's needs.

Teachers may also employ computers as an instructional aid. Computer-assisted instruction involves one or more students and a computer in drills, tutorials, games and simulation. Drill and practice programs are commonly used on computers in classrooms. Such programs provide students with additional practice in reviewing information previously taught by the teacher. Mathematics fact drills and
identification of the parts of speech are examples of drill and practice programs. Computers can also introduce new concepts to students. Tutorial programs sequence the unfamiliar skill or curriculum into small steps to be mastered in order. Students can advance in the instructional sequence by successfully completing posttests or be given immediate remedial instruction if the material is not understood. Computer graphics capabilities make simulation programs and games fascinating to young children. Simulations such as the working of a human heart help clarify processes which would otherwise be inaccessible to students. Other simulation programs, such as Apple's "Lemonade Stand," provide many opportunities for children to make decisions and consider the financial impact of variables like the weather, the price of lemons and sugar, and the prices of the competition on their profits. Computer-assisted instruction involves the use of a courseware program and a computer. Students may respond to the questions in the program, whether in the form of drill, tutorial, or simulation, but they may not alter the program itself.

The ability to use a computer and understand the basics of programming will become important to effective functioning in modern society. Some students are learning to program computers. They are commanding the computer to do what they want rather than simply responding to a preset program.
Seymour Papert, who studied with Jean Piaget for five years, advocates teaching students computer programming. In his book *Mindstorms*, he differentiates between computer-assisted instruction and computer programming. Papert (1980) notes that

... this difference is not a matter of a small and technical choice between two teaching strategies. It reflects a fundamental difference in educational philosophies. It reflects a difference in views on the nature of childhood. I believe that the computer as writing instrument offers children an opportunity to become more like adults, indeed like advanced professionals, in their relationship to their intellectual products and to themselves. (p. 31)

Students who are engaged in programming use the computer as a tool for their thinking, just as adults use a pen or pencil to work out their ideas. As they program the computer, they are "active builders of their own intellectual structures" (Papert, 1980, p. 19). They become masters of a very powerful technology which expresses their intellectual creativity. Students learn to reflect on their thinking and utilize problem-solving strategies as they study their own computer programs.

Microcomputers have the potential of transforming the school setting to a more creative, intellectually challenging environment. It depends on the way they are used in the classroom. Presently, the majority of students who have access to a computer are using computer-assisted instructional programs. In effect, the "computer is being used to program the child" (Papert, 1980, p. 5) as the child is
placed in the role of a passive respondent to the computer program. In a drill and practice environment, experimentation, creativity, and initiative have no place.

Papert, a mathematician at the Massachusetts Institute of Technology, has developed a computer language called LOGO which is simple enough for young children to learn, yet powerful and interesting to adults. LOGO combines structured programming with graphics. As a programming language, LOGO allows the user to be in control of the computer.

Students who program in LOGO can see immediately the effects of the commands they give the computer. An abstract object shaped like a triangle is used on the screen to draw the graphics. It is called the "Turtle," and it is helpful in spatial orientation on the screen and in thinking about programming. "The idea of programming is introduced through the metaphor of teaching the Turtle a new word (Papert, 1980, p. 12). For example, to write a program for a square, one could tell the Turtle by typing on the keyboard:

```plaintext
TO SQ
  >FD 100 moves the Turtle in a straight line a distance of 100 Turtle steps of about a millimeter each.
  >RT 90 causes the Turtle to pivot in place 90 degrees (Papert, 1980, p. 12).
  >FD 100
  >RT 90
  >FD 100
  >RT 90
  >FD 100
END
```

The computer would then show on the screen: SQ DEFINED. To have the Turtle make another square, SQ would be typed and
the Turtle would draw a square on the screen. The immediacy mode of graphics in LOGO is particularly suited to young children who are impatient to see their picture—the product of their programming efforts. Doing projects such as SQ is interactive learning. Children actively manipulate the Turtle which responds to their commands. Through experimentation in the LOGO environment, children discover important mathematical concepts and acquire a sense of control and powerfulness over the Turtle and the computer. They become computer programmers, not merely responders as in computer-assisted instruction.

Sixth-grade children programming in LOGO have been studied in Brookline, Massachusetts (Watt, 1979) and third-grade LOGO programmers have been studied in Dallas, Texas (Gorman, 1982). However, no studies reported at this time have been conducted with children younger than eight years of age who are programming in LOGO on a microcomputer. There is a need for research to be done with young children to determine their ability to program with a microcomputer. If Papert is correct in his belief that the learning of programming by children is a powerful tool for the development of intellectual and creative growth, then it is important to ascertain the appropriate level for beginning the teaching of programming. The intent of this study was to determine the feasibility and appropriateness of teaching five-year-old children to program a microcomputer using the LOGO language.
Statement of the Problem

The problem of this study was to describe and analyze the computer programming abilities and thought processes of five-year-old children using a conventional microcomputer.

Purpose of the Study

The purpose of this study was to analyze (1) the computer programming abilities of five-year-old children, (2) the conceptual, perceptual, and physical difficulties they encounter in using a conventional microcomputer, (3) the affective responses of young children to the computer, and (4) the thought processes of five-year-old children while programming the computer.

Research Questions

1. What do young children believe about how a computer works? What role do they think they have in the process?
2. What kinds of programming projects interest young children?
3. What styles of programming do the children exhibit?
4. When should concrete aids be introduced in explaining concepts? Of what value are they?
5. How long can the computer hold the attention of five-year-olds? Under what conditions does their attention span vary?
6. How do young children feel about working with the computer? What negative responses or feelings, if any, do they have toward working with it?

Significance of the Study

Presently most young children involved with a microcomputer are using it for reinforcement of skills in a drill and practice format. If children as young as the age of five are capable of learning to program and the outcomes appear beneficial, then curriculum decision makers may wish to consider allocating school time and resources for programming as well as for computer-assisted instruction in kindergarten and the primary grades.

Very little research exists about young children programming computers. Radia Perlman (1976) states that

there is a whole area of research involving working with children with the system, finding difficulties they encounter, and either inventing new ways to present the concepts they find difficult or deciding that there is some maturing the child needs in order to gain certain concepts. (p. 30)

There is a need for research which describes the process of young children learning to program a computer and their cognitive and affective responses to it.

Limitations of the Study

It was necessary to limit the size of the population to five students in order to analyze in detail the behavior, comments, and programming of each student. Thus the
generalizability of the results of the research is affected. The data collection was limited to one geographical area and to one racial and socioeconomic group.

The researcher performed the role of teacher and observer during the collection of data. Although friendly rapport was established with the students during the study, the researcher attempted to maintain an analytic viewpoint on their behavior. The constant use of an audio tape recorder compensated for any observational deficits in the researcher as instruction was in progress. The researcher also transcribed the tapes and analyzed the transcriptions. Thus a degree of subjectivity in the interpretation of the data may have occurred. However, the fact that these various roles were carried out by one person provides continuity and understanding to the interpretation of the data.

Definition of Terms

The following terms have restricted meaning and are thus defined for this study.

Computer refers to the Apple II Plus microcomputer.

Computer programming refers to the demonstrated ability to define a procedure, name it, save it, and recall it later by its name.

Dribble file refers to a detailed record of all commands typed on the computer.

Floppy disk refers to a flexible minidisk.
LOGO is the name of the computer programming language developed by Seymour Papert. It has been adapted and distributed by LOGO Computer Systems for use with the Apple II Plus microcomputer.

Monitor refers to the screen on which the commands and the graphics are displayed.

Procedure refers to a "list of statements considered as a unit, which act on data in a specified order" (Statz, 1973, p. 34). A procedure is created by the programmer to make the computer perform operations or commands which will accomplish the desired goal.

Subprocedure refers to a subset of the computer program. For example in a program designed to draw a face, each feature would be a subprocedure.

TO FACE
>HEAD
>EVES These are subprocedures. This is a printout of the subprocedure for HEAD.
>NOSE
> MOUTH
>END

TO HEAD
>REPEAT 360 [FD 1 RT 1]
>END

Procedures of the Study

Sample

Six students were randomly selected from a pool of ten kindergarten students who could identify letters and numerals out of sequence. The screening pretest was administered to all kindergartners attending the elementary school where
the research was conducted. A boy moved away during the study so the sample was reduced to five participants.

**Treatment**

Individual sessions were conducted daily with each of the children in the study. The sessions lasted about twenty minutes and were held in a vacant classroom containing an Apple II Plus microcomputer. The duration of the study was ten weeks.

**Collection of Data**

Several instruments were administered to the children at the beginning and at the end of the study. They included a test of number quantity, nine Piagetian tasks, and an informal student interview. Two programming tests were given to each child at the end of the study. Parents of the participants were asked to respond to a written questionnaire.

In addition to the test and interview results, data were generated in the daily sessions with the computer. Each session was tape recorded and transcribed. A dribble file of commands typed on the computer during each session was kept along with the researcher's field notes describing the child's behavior.
Analysis of Data

The transcribed data from the daily individual sessions with the computer were coded into five general categories of student behavior with the computer: (1) conceptual difficulties, (2) perceptual difficulties, (3) physical difficulties, (4) affective responses, and (5) thought processes. Each of these main categories was subsequently organized into subtopics.

The instruments administered at the beginning and at the conclusion of the study were analyzed. Student computer programs were examined and individual case studies were written.

Summary

This study was designed to provide a detailed description of the behavior of five-year-old children as they learned to program in LOGO. Of special interest were the conceptual and perceptual difficulties experienced by young children in relation to computing, their affective responses, and their thought processes as they interacted with the computer.

A review of the literature pertinent to these topics is found in Chapter II. The methodology and procedures of the study are described in Chapter III. A detailed analysis of data is located in Chapter IV. Chapter V includes implications, recommendations for further research, and a summary of the findings.
CHAPTER BIBLIOGRAPHY


CHAPTER II

REVIEW OF LITERATURE

The literature was reviewed concerning the topics stated in the purposes of the study: the computer programming abilities of children, the effects of computer programming on children, and the thought processes of young children. The review provided a comprehensive background of research that has been conducted with children and computers through 1982. A summary of research findings for each of the three topics follows.

The Computer Programming Abilities of Children

During the 1970s the National Science Foundation funded several research studies related to the abilities of children from the ages of three to fifteen to program computers. Most of these studies were conducted on university campuses and involved a minimum of sixteen subjects.

A. B. Cannara was the primary researcher for one of the NSF studies at Stanford University in 1973. He worked with forty-four children who were ten to fifteen years old using two computer languages, LOGO and SIMPER. Students received group instruction weekly and individual practice time on the computer the other four days for nine months.
Although all students studied LOGO and SIMPER, most students preferred programming in LOGO because it offered "easy access to message and picture processing" (Cannara, 1976, p. 68).

In 1977, Holzman and Glaser studied six, eleven-year-old boys as they learned to program in FOCAL or LOGO at the University of Pittsburgh (p. 5). Each student spent approximately twenty-four hours interacting with the computer during the six-month study. Three of the students were assigned to learn LOGO and the other three learned FOCAL.

The purpose of the study was to examine the types of programs children write and their programming difficulties in the two languages. The students became competent programmers and wrote a dice game, an airplane simulation game, and several computer-assisted instruction programs. Holzman and Glaser (1977) noted that the students' "only real obstacle to their progress was their lack of familiarity with the keyboards of their terminals" (p. 6).

The researchers concluded that both FOCAL and LOGO were appropriate languages for sixth-grade students. They recommended that "children a year or two younger would probably benefit more from a language like LOGO" because answers to arithmetic problems appear in integers and because words may be used as variable names (Holzman & Glaser, 1977, p. 11). The computer programming language FOCAL was said to be more appropriate for adolescents and adults than LOGO because of its advanced mathematical functions and subscripted variables.
The Brookline LOGO Project, supported in part by the National Science Foundation, was conducted at an elementary school in Brookline, Massachusetts for two months during 1977 and two months during 1978. The LOGO programs of sixteen sixth-graders were analyzed in detail to obtain information about student learning styles and programming abilities. Student participants, who represented a wide range of abilities including the exceptionally gifted and the learning disabled, worked on the computer for twenty to forty hours during the project. Seymour Papert (1978), the principal researcher, reported that all students, except two in the lowest quartile of national achievement scores, were able to program the computer independently (p. 1.15).

Students exhibited diverse programming styles. Some students were "top down" programmers who started with a clear idea of the desired end result of the program. Then they systematically wrote a program which would accomplish their objective. Others preferred to tinker around during programming and let the product emerge from the process. Such students were referred to as *bricoleurs*—a French word referring to those who "messed about" until they were satisfied with the result. Papert et al. (1978) indicated that "the style of bricolage may be much more natural and much more productive than is usually admitted" (p. 1.18).

Radia Perlman (1976) studied twenty-five preschoolers from the ages of three to five years as they encountered the
TORTIS (Toddler's Own Recursive Turtle Interpreter System) environment. The participants met individually with the researcher at the Artificial Intelligence Lab at MIT for an hour each week, for ten weeks. The conventional computer keyboard was replaced with a system of modular components which could be added as desired to increase complexity, but which eliminated the need for typing. The components, consisting of four button boxes and a "slot machine," could send commands to a small, circular, computer-controlled robot equipped with a light, horn, and pen or to a triangle on the display screen known as a "turtle." The commands could be sequenced on cards placed in the slot machine, but were not named or defined as procedures.

The children were able to manipulate the physical robot turtle and the display turtle, but they were not doing any formal programming in that they did not plan procedures. Perlman noted that four-year-olds had trouble distinguishing right and left and that they pushed direction buttons at random to make patterns.

While the children were delighted by all the stuff that got displayed on the screen they did not realize all those pictures were commands and steps in the procedure. While this was a lot of fun, it was more a magical mechanical process that somehow created nice results as opposed to being something the child knew he was really controlling. (Perlman, 1976, p. 16)

Doorly (1980), a teacher and coordinator of the gifted education program in Mansfield, Connecticut reported that
gifted children who were six to ten years old were successfully programming TRS-80 microcomputers in BASIC (p. 63). Children in grades one through four who had high scores on the Stanford Achievement Test in mathematical aptitudes and computation participated in eight-week computer programming units. Children were encouraged to complete a final computer science project which they designed according to their individual interests. Some children chose to write programs for new games. Doorly emphasized the appropriateness of the study of computer science by children and 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).

Cynthia Solomon and Seymour Papert (1976) conducted a follow-up case study of a second-grade child who had been taught simple programming in LOGO six months before as a first-grader. They recorded her programming and the problem-solving strategies she employed when she worked on a main frame computer for several hours during a weekend. She made stick figures and a row of flowers on the screen. She created a paper sign to help her to remember the order of letters in the name of her program (in order to prevent reversals). The researchers noted that the child preferred to use RT 90 repeatedly instead of using LT 90. This
particular child was able to program in LOGO when she was a first-grader. Solomon and Papert observed that she was a much more independent programmer six months later.

Summary of Literature Concerning the Computer Programming Abilities of Children

All of the research studies reviewed involved children learning to program in LOGO or a derivative such as TORTIS, except for the work of Doorly's students in BASIC. The available research suggests that LOGO is an appropriate programming language for children.

All the LOGO studies cited used a mainframe computer, rather than a microcomputer. The mainframe provided additional power to operate interesting peripheral devices such as a computer-controlled robot which could draw on paper, flash lights, and make sounds. A robot has been developed which is powered by a microcomputer. The cost is prohibitive at present, however, and few sites have a robot available. This study used a conventional microcomputer and a silent Turtle on the display screen to determine whether young children could program using unaltered hardware.

One other factor that all of the studies reviewed had in common is that they were all qualitative in design rather than quantitative. The research designs were not empirical, and they did not use statistical models. Rather, they were
descriptive in nature and reported case studies of individual students' work. A. B. Cannara (1976) stated that

protocols and (tutorial notes) are precisely the data upon which [his] work is founded. The primary objective is to understand how children learn programming concepts. With error-analysis as a tool, student/machine interactions must be exposed in as much detail as possible. (p. 14)

Seymour Papert also scrutinized the child-computer interactions by analyzing the "dribble file" of each student's daily work on the computer. He defined the dribble file as being the "complete printed record, key stroke by key stroke, of interaction with the computer" (Papert et al., 1978, p. 1.9). In addition, the teacher made daily anecdotal records of each student's work. The instruments used in these qualitative studies were devised by the researchers and included questionnaires and informal pretests and posttests.

A review of the literature concerning children programming computers reveals that children who are six years old (Doorly, 1980 and Solomon & Papert, 1976) and older have demonstrated the ability to program computers. Although Perlman (1976) worked with preschool children, she discovered that they were unable to plan and carry out a procedure systematically, even though many external system adaptations were made to accommodate the abilities of young children.

One question which remains unanswered is: What are the computer programming abilities of five-year-olds working
The Effects of Programming on Children

A justifiable concern about children programming computers is: How does it affect their lives? A few studies have been conducted to determine the cognitive effects of computer programming upon children.

Henry Gorman (1982) studied fifteen third-grade LOGO programmers at Lamplighter School in Dallas, Texas. One group of students received LOGO instruction for one-half hour per week and the other group received one hour of LOGO instruction per week. Gorman reported that "after nine months, the students who studied LOGO for one hour were significantly better on a rule-learning task than their classmates who were given one-half hour of LOGO instruction a week" (1982, p. 1). Gorman questioned whether the

... superiority in rule learning of the children with the extra LOGO training represented merely a specific gain in classification of stimuli into conceptual or perceptual categories, or [whether it was] representative of a broader general problem solving schema acquired through LOGO. (1982, p. 7)

Another researcher who investigated the relationship between children's problem-solving abilities and their development of computer programming concepts was Joyce Ann Statz. Her study at Syracuse University in 1973 involved twenty ten-year-olds learning to program in LOGO matched
with a control group of non-programmers. After the children were administered problem-solving tasks, the researcher concluded that the children who were in the LOGO group performed better on the permutation tasks and word puzzle problems. However, on the whole, the LOGO programmers were not better problem solvers than the control group (Statz, 1973, p. 181).

Some case studies of student programmers indicate that learning to program a computer has a positive effect on student self-concepts and their desire to interact with other students. In the final report of the Brookline LOGO project (1979), Dan Watt described the affective aspects of a sixth grader's experience with LOGO. At the beginning of the project, a severely learning disabled student named Karl had few friends and was passive and noncommittal in his attitude toward school. The following sentences, excerpted from the case study, describe the affective changes in Karl as he learned to program in LOGO. Karl learned to program and "developed an experimental approach to the system, by trying things and seeing what happened" (Watt, 1979, p. 10.8). As his success continued, Karl became more assertive and curious. He asked what error messages meant, and sought to understand how to use new commands (p. 10.11).

At about the same time, Karl began to express an interest in the work of other children in his group, and began to show them his work. He invited a friend to class and swapped programs with him. He began to show that he was feeling good and enjoying himself. His face was more expressive, his posture more relaxed.
Changes in Karl's attitude toward his classroom work were also noted by his regular teacher. After about twenty classes she reported that Karl was beginning to show that he really cared about his schoolwork, that he had begun concentrating on his work in a way that she had not seen before, and that he seemed to have a great deal more confidence in his ability to carry out academic tasks. She attributed these changes to his feeling of success in the LOGO classes. (p. 10.12)

Karl's competence as a programmer gave him new confidence in relating to other students. It also renewed his confidence in his ability to succeed in his other schoolwork.

Karl initiated his own computer programming projects. He was actively engaged in accomplishing his own purposes as he programmed in LOGO. This experience was "ego syntonic" for him. This Freudian term means that the "ideas are acceptable to the ego and compatible with the ego's integrity and with its demands" (Papert, 1980, p. 221). Seymour Papert has borrowed this concept and has applied it to the experience children have when they learn to program computers. Programming is "ego syntonic in that it is coherent with children's sense of themselves as people with intentions, goals, likes, and dislikes. A child who draws a Turtle circle wants to draw the circle; doing it produces pride and excitement" (Papert, 1980, p. 63). According to the available literature, learning to program computers can be a meaningful learning experience for children because it requires their interest, initiative, and creativity. Watt's study indicated that
computer programming may produce positive feelings toward self and others in some cases.

Summary of Literature Concerning the Effects of Programming on Children

How does computer programming affect the lives of children? Research suggests that in the cognitive realm, programming helps children classify things systematically and form rules or generalizations from the categories they create. It also improves their performance on some problem-solving tasks. Although the evidence is very limited at present, there is some reason to believe that the experience of learning to program a computer may have a positive effect on a child's self-concept. This feeling of competence and confidence may transfer to the child's interactions with others and to other responsibilities.

The Thought Processes of Young Children

Since one of the stated purposes of the study was to analyze the thought processes of five-year-old children as they were engaged in computer programming, the literature was surveyed to obtain information about their thinking and about any developmental idiosyncracies in thinking or reasoning they typically exhibit while involved in problem solving.

The literature reviewed was written from a Piagetian perspective. John Flavell (1977) differentiates between the
two primary theoretical conceptions of the cognitive system in his book, *Cognitive Development*.

One approach might be termed the **structuralist-organismic** orientation, chiefly represented by Jean Piaget's theory of the mind. The other is the information-processing approach, as embodied in the writings of Herbert Simon, Allen Newell, Walter Reitman, and numerous others. (p. 5)

Although both approaches offer credible and respected explanations of cognitive functioning, the perspective of the structuralist-organismic approach was adopted for this study because the subjects were young children. The work of Piaget and his colleagues in observing and recording the behavior of children of various ages who were confronted with problems of causality was influential in the interpretation of the behavior of the children in this study. Children faced with such tasks have an opportunity to become aware of the results of their actions upon objects. The five-year-olds in this study were also experiencing causality as they worked with the computer. They caused the Turtle to move on the monitor by giving it commands, and they attempted to explain the actions of the Turtle. The thought processes of the five-year-olds in the study were analyzed as they were revealed through their questions, comments, and actions.

**The Piagetian View of Cognitive Functioning**

The Piagetian view of cognitive functioning is complex, comprehensive, and addresses epistemological problems as well
as psychological ones. The following summary of his position is necessarily limited to that which concerns the thought processes of five-year-olds.

According to Piaget, the child constructs knowledge as he interacts with the environment. He "actively selects and interprets environmental information . . . to make it fit in with his own existing mental framework" (Flavell, 1977, p. 6). This is accomplished through the processes of accommodation and assimilation. Accommodation occurs when a person takes in information from the environment. Assimilation follows as that information is interpreted and incorporated into what the person already knows. "Therefore the cognitive system simultaneously adapts reality to its own structure (assimilation) and adapts itself to the structure of the environment (accommodation)" (Flavell, 1977, p. 13). This has been termed an interactionist model of cognitive functioning because of the constant interchange between the subject and the environment.

Characteristics of Preoperational Thought

This model is also composed of developmental stages through which individuals pass as they mature. Most five-year-old children exhibit the behavioral characteristics of the preoperational stage of Piaget's sequential theory. Children in this stage are able to communicate well, verbally
and nonverbally. They are beginning to develop self-control, and they are becoming aware of "consistencies, invariants, and regularities in the world" (Flavell, 1977, p. 78). Their thinking possesses a sort of partial logic as they have certain notions about the permanence of identities and about functional relationships, but cannot yet reverse operations.

For example, a child in the preoperational stage believes that the amount of water changes as it is poured from one glass into a taller, thinner glass in Piaget's conservation of liquid task. However, the child asserts that it is the same water being poured, thus retaining the essential identity of the water though the appearance of it has changed. Flavell (1977) notes that

For this particular physical transformation, then, Piaget would say that our child possesses a type of qualitative invariant called identity but has not yet acquired a markedly different, quantitative invariant called conservation of amount. As such, his behavior reflects preoperational rather than concrete-operational logic. (p. 74)

The way preoperational children perceive functional relationships is also qualitative rather than quantitative in nature. Flavell (1977) explains that

In mathematics, x is said to be a function of y, i.e. $x = f(y)$, if the value of x depends upon and covaries with the value of y. The child of this age is able to note simple functional relationships and recurrent covariations among observable events. (p. 76)

He is able to relate changes in one thing to changes in another.
In an experiment designed to allow children to formulate functional relationships, the thinking of preoperational children was made evident (Piaget, Grize, Szeminska, and Vinh Bang, 1968, pp. 50-57). When five-year-old children were presented with the task of feeding three toy fish of differing sizes a proportionate amount of food, they intuitively varied the portions given depending on the size of the fish. The amounts, though, were not quantitatively proportional (according to the directions of the task which stipulated that the middle-sized fish be given twice as much food as the smallest fish.) But the children did give some amount more to fish according to size even though their conception of the functional relationship was not a precise, quantitative one.

The attainment of identity constancy and functional relationships is an important cognitive achievement. The preoperational child perceives the regularities in the environment as he or she attempts to sort and order it. Flavell (1977) observes that

Early childhood is the period when regularities first become objects of representational rather than sensory-motor knowledge, and the fact that the child's initial representation of them is qualitative rather than quantitative in character hardly detracts from its significance as a cognitive developmental milestone. (p. 78)

**Conceptualization and Cognizance**

Piaget identifies three developmental levels of knowledge: (1) material action without conceptualization (occurs
during the sensorimotor stage), (2) conceptualization, and (3) reflected abstraction (occurs during the stage of formal operations beginning at age eleven or twelve) (Piaget, 1974/1976, p. 349). Each level is built upon what was learned in the previous level of knowledge. Five-year-olds are engaged in conceptualization based upon the action schemes of Level I. Their understanding of the relation of actions upon objects evolves gradually (Piaget, 1974/1976, p. 327). As they mature, children understand why their actions have certain effects. They know how they have succeeded and can explain the interaction (Briquier, 1977/1980, p. 90).

The process of becoming aware of physical actions and their causes is a form of conceptualization (Sinclair, 1978, p. 196). Hermione Sinclair, a colleague of Piaget, describes some experiments dealing with causality which encourage children to formulate explanations of physical actions and verbalize their thinking.

The experiments concern situations where the child is asked to perform an action (usually with a few objects) whose result is clear to him (e.g., to use a sling to get an object into a box, to propel a ping-pong ball on a table in such a way that it returns to its starting point, etc.). Sometimes the child succeeds immediately without any trial and error, and the experimenter asks questions ("How did you do it? How did it work?") inciting the child to verbalize or to show in slow motion how he succeeded; usually, awareness in this sense lags considerably behind success in action. Sometimes the child does not immediately obtain the desired result, so he has to stop and think; then his awareness of the difficulty and, possibly, of a way of overcoming it can be inferred not only from his answers to questions but also from his actions. Clearly, the
experiments . . . deal with the child's growing awareness of his own actions on objects and of the interactions between objects that may result from them (e.g., when one sets up a series of dominoes in a certain way, a tap on the first one will make all the others fall down one by one). (Sinclair, 1978, p. 192)

These experiments are valuable in that they provide children opportunities for conceptualization—of thinking about and describing physical actions and their causes.

Young children may also express their understanding of actions through nonverbal modes. Piaget (1974/1976) believes that "conceptualization is possible both outside language and linked with the other forms of the semiotic function (mental image, for instance and drawing)" (p. 328). The preoperational child represents goals through mental or written images and through internal language or verbalized speech.

Piaget (1974/1976) views cognizance as a process which begins with consciousness of a goal (mediated through representation in children), action is taken to achieve the goal, and the result is evaluated for success or failure (p. 334). Young children typically use the trial and error approach to solving a problem. They vary their actions haphazardly without understanding because on some tasks their "conceptual interpretation still lags behind the action" until the age of seven (Piaget, 1974/1978, p. 216). There are times, however, when "the action and its conceptualization are almost on the same level and when there are constant exchanges between them" (Piaget, 1974/1978, p. 217). This process
involves active adjustment on the part of the subject in the pursuit of the goal. "Conceptualization . . . provides action with limited and provisional plans that have to be revised and adjusted" during the preoperational period (Piaget, 1974/1978, p. 217). The child consciously considers options and choices.

**Success and Understanding**

According to Piaget (1974/1976), when a young child obtains his goal, he is successful. However, in the case of failure, the child "turns his attention to the means used and to how he might correct or perhaps replace them" (p. 335). Piaget (1974/1978) differentiates between the achievement of a goal (success) and understanding how the goal was accomplished.

Success means having enough understanding of a situation to attain the requisite ends in action, and understanding is successful mastery in thought of the same situation to the point of being able to solve the problem of the "how" and the "why" of the connections observed and applied in action. (p. 218) In short, understanding brings out the reason of things, while success is simply their effective utilization. (p. 222)

Young children are often successful in accomplishing their desired goal through a series of attempts, yet they are unaware of exactly how it was achieved. In such instances they are successful, but without understanding. Piaget (1974/1976) records that, in his research, many children exhibited such behavior.
The scheme that assigns a goal to the action immediately triggers off the means of effecting it (regardless of how appropriate these may be) may remain unconscious, as is shown by the multiple situations studied in this book where the child achieves his goal without knowing how he did so. (p. 334)

He notes that "success in specific actions of a causal nature generally precedes cognizance of them" (Piaget, 1974/1976, p. 300). Piaget (1974/1978) attributes the success of cognitive powers in action without understanding to cybernetic causality, deriving its powers of organization from "loops." In other words, the movements constituting actions do not follow one another in a straight line but are joined together into the relatively closed cycles of schemata, and the latter are teleonomic (goal-directed). These schemata are conserved by their very deployment; they fit the objects they use into these cycles, and this is the process of cognitive assimilation. (p. 220)

The young child unconsciously selects from his repertoire of action schemes those which are appropriate to the goal and is able to solve the problem without knowing how he did so.

**Problem-Solving Behaviors of Preoperational Children**

There are some specific behaviors demonstrated by young children as they are presented with problems of causality. One such behavior that is typical of the preoperational child is incomplete explanations of the physical actions occurring in the causality task. The description of his own actions and the interaction between objects is sketchy and there is
difficulty in expressing relationships. Piaget (1974/1976) states that

this is what we observed on several occasions when the child was asked how he came to discover a specific process. While the young subjects merely recounted their successive actions (or at the beginning merely represented them with gestures and no verbalization), the older children said, for instance, "I saw that... so I thought... or so I had the idea..." and so on. (p. 377)

Again, the young child focuses on actions and not on the options considered or the reasons certain ones were selected.

Another area important to the consideration of young children's thinking during problem solving is their reaction when the result of their action does not bring about their desired goal. There are a number of possible responses from preoperational children in such a discrepant situation: (1) nonrecognition of the failure of the action, (2) denial of responsibility for the failure, or (3) acceptance of the failure. Children may exhibit any of these responses, depending upon the task with which they are confronted.

The first response, nonrecognition of failure, is peculiar to young children. Piaget (1974/1976) notes that when Level IA subjects (four- or five-year-old children) are faced with a conflict between a construction and a previous scheme, "they simply distort their observations and seem to drive the source of the conflict back into the unconscious" (p. 339). A child of this age tends to see the affirmative aspects of the result and ignores any discrepant elements.
Since he is unable to reverse operations, he has difficulty spotting the negative elements of a contradiction. "The child's preconceived ideas influence his reading of the situation--that is, he sees what he thinks he ought to see" (Piaget, 1974/1976, p. 300).

The second response which young children give during the failure of the result to meet the goal is rejection of any responsibility for the result. Children who respond in this way acknowledge that the outcome of the actions does not conform to the original goal, but deny that they caused the erroneous result. They attribute the failure to the resistance of the objects involved in the causality task (i.e. the pencil caused the mistake) (Piaget, 1974/1980, p. 93). This denial of responsibility is a sign of ontogenetic maturity, however, because the contradiction is recognized.

The third response to a discrepancy between a goal and its result is recognition and acceptance of the contradiction. "The negation is attributed to the actions of the subject," and this response is the most advanced in "the progressive interiorization of negation" that occurs in the child (Piaget, 1974/1980, p. 93). Once the child recognizes that the goal and the result are not the same and accepts responsibility for it, he or she is faced with two alternative courses: (1) change the original goal to conform to
the result obtained or (2) attempt to change the result through additional actions to achieve the desired goal. Either choice is acceptable in the process of equilibration which is characterized by "direction without finalism" (Piaget, 1974/1978, p. 228). The young child deals with conflict in a way that produces a satisfactory result and continues the process of representing another goal and searching for the means of realizing it.

Summary of Literature Concerning the Thought Processes of Children

The Piagetian model of cognitive functioning views young children as being actively involved in constructing knowledge of the world. Through the processes of accommodation and assimilation, children select information about the environment and incorporate it into what has previously been learned.

Five-year-old children are preoperational thinkers according to Jean Piaget's model. They are not able to reverse operations, but they do have some qualitative notions of the constancy of identity and functional relationships. Young children can think about performing actions in the future with the help of some form of representation—either by means of gestures or drawings or through internal or verbalized language. This process aids in defining exactly what they want to accomplish. Once their goals are finalized, they immediately set out to achieve them through action.
Following the action, if the result matches the goal, a successful conclusion is reached. If the result differs from the goal, preoperational children may not recognize the fact that a discrepancy exists. In some instances they may recognize the contradiction but reject any responsibility for its occurrence. Or they may accept the differing result and attempt through trial and error to change it to fit the original goal. Sometimes they are satisfied with the unexpected result and decide to modify the goal to conform to the novel result.

Young children may not be able to express what they are thinking in a comprehensive way. They may explain their actions through gestures or by performing the actions again. Verbal descriptions are incomplete.

Five-year-old children may not be able to explain their actions on physical objects because they are not aware of the relations involved in the task. They may be able to perform a causality task correctly, yet be unaware or unconscious of why it worked. In such cases, Piaget asserts that success precedes understanding.

The preceding information about the thought processes of preoperational five-year-old children provides a perspective for understanding their actions, language, and thinking while they are engaged in computer programming.
CHAPTER BIBLIOGRAPHY


CHAPTER III

METHODOLOGY

The purpose of Chapter III is to describe how the study
was conducted. The following aspects are discussed: the
selection of student participants, the instruments adminis-
tered during the study, the procedures used to teach program-
ing to the five-year-olds, and the procedures for the
collection and analysis of the data. The first section of
this chapter presents the design and findings of the pilot
study.

The Pilot Study

A pilot study was undertaken prior to the collection of
data for the present major study because little information
was available about procedures which might be used to teach
young children programming. The purposes of the pilot study
included (1) ascertaining which concepts were difficult for
five-year-olds, (2) developing strategies and aids for
teaching children LOGO, (3) determining the appropriateness
of the pretests and posttests, (4) experience keeping a
dribble file and taking notes during individual sessions,
and (5) testing various placements of the tape recorder to
obtain quality sound production of the interactions at the
computer and around the room.
Sample and Setting

Five five-year-old children who spoke only English were enrolled at the North Texas State University nursery school during the Summer term of 1982. Permission was obtained for all five of them (three girls and two boys) to participate in the pilot study. The study was conducted for four weeks during June and July of 1982. The participants met individually with the researcher for about twenty minutes every day from Monday through Friday. An Apple II Plus computer equipped with the Apple LOGO language was set up in the "rest area" which was located in a room adjoining the activity area of the nursery school.

Instrumentation

Nine Piagetian tasks were administered prior to the introduction of Apple LOGO. They were given only once because of the brief duration of the pilot study and the time required to administer them individually (about a week).

A short test developed by the researcher was given as a pretest and posttest. It tested numeral and shape recognition, knowledge of number quality and concepts of directionality (right, left, forward, and backward movement).

A student questionnaire was also devised. Informal student interviews were conducted at the beginning and at the conclusion of the pilot study.
At the end of the study, the students were asked to complete a task used in the Brookline LOGO Project (1978). It was designed to test directionality and programming skills. The participants were asked to draw a line on paper from the starting point to the end of a simple maze and describe the number of blocks passed and the directions they took.

**Teaching Procedures**

Five-year-old children often confuse the directions of right and left. In an effort to alleviate the confusion, the Apple LOGO abbreviations of RT (right) and LT (left) were written on blank labels which were placed on the backs of the students' hands.

At the beginning of the pilot study, time was devoted to experiencing body movement in space. The student and teacher took turns at issuing commands such as "Turn to the right; move backward six steps" while the other responded. A square was placed on the floor with masking tape. As students walked on the sides of it, they counted their steps and announced the direction of turning at the corners. The researcher recorded the commands on the chalkboard. Then students typed the commands into the computer.

Participants played a game on the computer which required the correct turning of the Turtle and a distance estimate to make the Turtle move to its box on the screen.
Labels were placed over the keys F, D, B, and K on the keyboard to reduce perceptual confusion and help students locate the keys faster. They used those keys when they gave the Turtle commands such as FD 60 or BK 30.

Data Collection Procedures

A tape recorder was used to record the administration of the instruments and some of the daily sessions. Various locations were tried in the room in an effort to get the optimal sound quality on the tape as students moved around the room and worked at the computer. The final arrangement consisted of placing the tape recorder on a portable cart next to the computer table, but out of reach of the participants. An external microphone was placed to the left of the computer. When the participants moved away from the table, the microphone could be placed on the cart and moved near the center of the room.

A dribble file was kept of the commands typed into the computer and the seemingly significant verbal interactions between the researcher and the participants were noted. No attempt was made to record every interchange that occurred during each session. Notes accompanied the file explaining the reactions of the children and a description of their behavior. Written records were kept of the results of the pretests and posttests.
Information from the Pilot Study

One of the purposes of the pilot study was to determine the concepts involved in programming which were difficult for five-year-old children. They are enumerated below.

1. Numeral identification—The pretest revealed that students could not name many one-digit numerals (especially 6, 7, 8, and 9) and only one of the five children could correctly identify the numerals 25 and 45. Due to the amount of time spent daily learning to identify numerals in the pilot study, it was decided to require proficiency in numeral recognition as a prerequisite skill for the major programming study.

2. Letter recognition—Participants had difficulty identifying letters out of alphabetical sequence on the computer keyboard. This problem had an adverse effect on the programming success of the participants. Typing was a slow, arduous process when all of the keys had to be searched one-by-one for the desired letter. Students often asked the researcher where certain letters were located. Some assistance was provided when it seemed that students were frustrated. There was a long wait time between deciding what command to type in and seeing its result on the screen. Because of the preceding reasons, another section was added to the pretest: letter identification. Participants in the major study were required to demonstrate the
ability to identify the capital letters of the alphabet out of sequence.

3. Sound-symbol confusion—Students frequently confused the symbol and a similar-sounding name for a numeral. They would say "thirteen" when referring to the numeral 30 and "sixteen" when they meant 60. The sounds of the letters D and T were also confused.

4. Left-right confusion—The children participating in the pilot study continued to confuse their right and left hands throughout the study. Because this behavior is typical of five-year-olds, labels were worn on both hands to designate right and left in the pilot study and in the major study.

5. Number quantity—Most students had no clear concept of number quantity before the pilot study began. Three of the participants were absent when the posttest was given. Therefore, it is not possible to determine whether a better understanding of the number quantity was an outcome of the pilot study.

Conclusions of the Pilot Study

The experience of working with five-year-olds and LOGO during the pilot study resulted in the revision of some procedures in the major study. These changes are outlined below.
Participants found the preprogrammed computer "Game" too frustrating. When they played it, they had one opportunity to estimate the numerical value of the distance of the Turtle from its box before the game generated another random position for the Turtle and box on the monitor. Most students failed at this more than they succeeded and wanted to stop playing after a few trials. A simpler version of "Game," ("G"), was introduced at the beginning of the major study.

Some of the instruments were altered as a result of their use in the pilot study. The student interview was changed for the major study to include three questions about emotions students had when they worked with the computer. In addition they were asked how the pictures got on the computer's screen in the revised interview. The maze task, borrowed from the Brookline LOGO Project (1978), was too difficult for five-year-olds. It was eliminated from the major study, and the researcher devised other tests of programming.

As a result of the pilot study, plans for working with the children and the computer at regularly scheduled times were abandoned. In effect, the participants taught the researcher about the optimal times for their working with the computer. The researcher observed the involvement of the participants in the activity centers of the nursery school. Students were approached to work with the computer
when they were looking for a center in which to work or when they were not fully involved in a center. This informal guideline was developed after participants refused to leave an extremely interesting center (such as face painting or water play) or pretended not to notice the researcher as she came near a child's favorite center. It was discovered that some children cooperated better and accomplished more on the computer when their individual session was held early, before they became fatigued and more distractible. Working on the computer was not as attractive to the participants as playing outdoors with their friends or playing in the water sprinkler. Therefore it was arranged for the individual sessions to occur only during center time in the major study so that students could participate in the group discussions and recess.

The pilot study provided information and experience essential to the design and execution of the subsequent major study. A description of the methodology of the major study follows.

The Major Study

Sample and Setting

The study took place at a public school in a city with a population of 52,000 located in North Texas. The school was situated in a predominately white, middle-class neighborhood. It was selected as the research site because
the principal had some experience in programming and was eager to use computers in the elementary school. During the time in which the study was conducted, grades three through six were learning to program in BASIC and were involved in computer-assisted instruction. Kindergarten students did not have access to computer use at that time.

The students who participated in the study attended kindergarten for one-half day daily at the public school. They were all five years old at the beginning of the study on September 13, 1982. The duration of the study was ten weeks.

The population of the study was selected by administering a screening test of numeral and letter identification to all kindergartners enrolled at the elementary school. Ten students were eligible to participate in the study based on the results of the pretest. Their names were classified by sex and by attendance in the morning or afternoon kindergarten. There were more eligible boys than could participate so their names were randomly selected. Three boys and three girls were selected as participants. Three of them attended kindergarten in the morning and three attended in the afternoon. Since one of the male participants moved away during the third week of the study, this record reports on the five remaining participants: two boys and three girls. All of the subjects were white. Parental permission was obtained for the children who participated in the study.
(See Appendix A.) Table I presents information about the sex, birthdate, chronological age, and absence of the five participants.

**TABLE I**

**SEX, BIRTHDATE, AGE, AND ABSENCE OF THE SUBJECTS**

<table>
<thead>
<tr>
<th>Name</th>
<th>Sex</th>
<th>Birthdate</th>
<th>Age</th>
<th>Absence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alexandra</td>
<td>F</td>
<td>1-10-77</td>
<td>5 years, 8 months</td>
<td>2 days</td>
</tr>
<tr>
<td>Andrea</td>
<td>F</td>
<td>9-20-76</td>
<td>5 years, 11 months</td>
<td>4 days</td>
</tr>
<tr>
<td>Billy</td>
<td>M</td>
<td>4-28-77</td>
<td>5 years, 5 months</td>
<td>2 days</td>
</tr>
<tr>
<td>Danielle</td>
<td>F</td>
<td>12-14-76</td>
<td>5 years, 9 months</td>
<td>0 days</td>
</tr>
<tr>
<td>Jonathan</td>
<td>M</td>
<td>10-3-76</td>
<td>5 years, 11 months</td>
<td>3 days</td>
</tr>
</tbody>
</table>

Individual sessions were conducted daily with each of the participants and an Apple II Plus microcomputer equipped with the Apple LOGO language. The sessions lasted from twenty to thirty minutes and were held in a vacant classroom to facilitate the quality of sound production on the tape recordings.

Records were kept of student absences. One student had perfect attendance. Two students were absent twice during the study. One student was absent three days, and one was absent four days. There was no excessive absenteeism during the ten-week study.
Instrumentation

A review of research pertaining to computer programming abilities of children was conducted to ascertain the kinds of instruments that had been previously used. No instruments were located by that review which were appropriate for use with five-year-old children. Therefore, the researcher formulated some simple instruments to determine entry level skills and computer programming competence at the conclusion of the study.

Screening test.—A short screening test was administered verbally using flashcards since most of the participants were not able to read. The test diagnosed the abilities of the students to identify selected numerals and letters. It also measured their understanding of the relative value of numbers, their understanding of spatial relationships, and their recognition of shapes. (See Appendix B.)

Piagetian tasks.—A number of Piagetian tasks were also administered at the beginning and at the conclusion of the study. The tasks selected for administration related to the concepts involved in programming and indicated whether the participants were thinking at a preoperational or concrete operational level of cognitive development according to Piaget's model. The following tasks were given to the participants: (1) conservation of substance, (2) conservation
of number, (3) conservation of liquids, (4) conservation of length, (5) conservation of distance, (6) conservation of area, (7) classification of geometric objects, (8) concept of seriation, and (9) ability to draw shapes with topological and Euclidean properties. Charting Intellectual Development: A Practical Guide to Piagetian Tasks (Formanek & Gurian, 1976) was used as a manual for the correct administration of the tasks and the evaluation of the participants' responses.

Student interview.—In order to determine the children's affective responses to the computer and their beliefs about their role in its operation, an informal interview was conducted at the beginning and at the end of the study. The interview questions were based on the ones used in the pilot study conducted in June-July, 1982. (See Appendix C.)

Parent questionnaire.—A written questionnaire was sent home with the participants to be completed by their parents. Parents were asked to relate comments that their children had made at home about their work with the computer. They were also asked to describe any behavioral changes that had occurred in the play, writing, or drawing activities of their children since they started working with the computer. Finally, parents were asked to evaluate how their children felt about working with the computer. (See Appendix D.)
Programming tasks.—At the end of the study, a number of programming posttests were administered. Students were challenged to connect dots on the screen by giving the correct commands. There were four dot tasks that increased in complexity. (See Appendix E.) Students were also asked to write procedures for three different shapes without the help of the monitor. The procedures were written on paper first and then checked out on the monitor. Revisions were made, if needed.

Teaching Procedures

Preskill games.—Three games were devised by the researcher (from information contained in the Apple LOGO manuals) to introduce the participants to the computer in a non-threatening way. These preprogrammed games were called "preskill games" because they were designed to introduce students to the prerequisite skills needed for programming in LOGO. They helped students learn to manipulate the Turtle spatially, learn the correct terminology for directions, and learn about number quantity. Students practiced typing numerals in sequence from left to right, they became familiar with the placement of keys which were used frequently in LOGO, and they learned to use the delete key to erase typing errors. The three preskill games are described below.
Students were introduced to LOGO graphics from the beginning in a game called "Draw." This game placed the Turtle in the middle of the screen. The child could draw with the Turtle by typing F for forward, B for back, R for right, or L for left. A series of Fs would make a line. This was a rather unstructured way of letting the child control the Turtle and create original designs. It also provided practice in manipulating the Turtle in space.

Another game introduced during the first week of the study was called "G." In "G," a square and the Turtle appeared on the screen. Their positions were randomly selected. The goal of the game was to get the Turtle inside its box. The child typed F, B, R, or L to move the Turtle a short distance toward its box. These keys were pushed repeatedly until the goal was achieved.

A more difficult version of this task called "Game" required the child to orient the Turtle so that its head was pointing to the box. Then a number input was given and the Turtle traveled forward that number of steps. A new random placement of the Turtle in relation to the box was subsequently generated.

Body orientation.—Students often confused the directions of right and left. During the study they wore labels on the backs of their hands indicating LT and RT. When the Turtle on the monitor was facing a direction other than
straight up, students would align their bodies with the Turtle and look at the labels on their hands before typing in a turning command.

At the beginning of the study, games were played with the children to help them orient their bodies in space. Students were asked to "be the Turtle." As the researcher issued commands such as FORWARD 10, BACK 3, the students responded taking ten small steps forward and three steps backward. Then the roles were reversed and the researcher assumed the role of the Turtle while the students gave directional orders.

**Drawings.**—Students were asked to draw their designs and pictures on paper before any attempt was made to program them. They decided what part of the picture they wanted to make first and where they wanted the Turtle to start the picture. Then the picture was drawn on a very large scale outside on the concrete playground with chalk. Students walked on top of the lines they drew and counted the number of Turtle steps aloud. The researcher walked beside the participants and recorded the commands and the number of steps as they were dictated. The commands were then typed on the computer to create the picture on the monitor. After several programs were drawn on a large scale outside, students had learned enough about number quantity to program their pictures from the paper drawings only.
Number quantity.--A number line was located on the wall above the computer. It listed the numerals from zero to 200 in increments of ten. When students were programming inside using their drawings as guides, they frequently referred to the number line for ideas for number inputs.

The number line was also used as an aid for teaching about number quantity. When a student tried a number that did not move the Turtle the desired distance, that number was located on the number line and a smaller or larger number was selected for the next trial.

At the beginning of the study, the researcher labeled one-digit numerals as "little" and numerals of 100 or more as "big." This practice of labeling numbers was employed in an experiment conducted by R. S. Siegler (1983) in which the subjects "attached a semantic code [small, medium, big] to each stimulus being compared" (p. 73). The following excerpt is from notes recorded during a session with a student during the first week of computer use. It illustrates the verbal labeling of numbers and the use of the number line as a reference for quantity. The student (ST) had just tried the number nine.

ST    Oh!
R    What happened?
ST    It only took little steps.
R    See this number line up here that I made? (counting) Six, seven, eight, nine and then there are some bigger numbers. The farther you go down the number line, the bigger the numbers are. So if you have a box here and a Turtle way over there (on the monitor), you choose one of those big numbers.
ST    Like 100.
In some cases the number line posted above the computer was insufficient, especially when students needed to compare numerals of similar sizes. R. S. Moyer and T. K. Landauer (1967) measured the time required for judgments of numerical inequality. They discovered that their subjects could compare the magnitudes of digits of discrepant sizes (thirteen and fifty) more quickly than they could compare the magnitudes of digits of smaller sizes (thirteen and fourteen) (p. 1520). In instances when students were comparing numbers in between the increments of ten posted on the number line, a special number line was written on the chalkboard. Here are two examples of number lines from the research notes. (Students tried the numerals underlined below as inputs.)

**EX 1.** 13 14 15 16 17 18 19 20
   too too ↑ ↑ too
   little little big

**EX 2.** 6 7 8 9 10 11 12

In example one the student tried two inputs that made the curve in her design too small. She found that twenty made it too large and decided to try seventeen or eighteen as inputs on the next day. The student used verbal labeling of the results. These labels were recorded under the number inputs. In example two the student tried six first, then eight, and was finally satisfied with twelve as an input. These special number lines were helpful in visualizing the
magnitudes of similar numbers. Although the five-year-old children in the study could recite the numerals in order, they needed the visual cue of a number line to aid in their understanding of number space and to help them remember which numbers had been tried as inputs.

**Data Collection Procedures**

Data were generated by the tasks described in the previous section entitled "Instrumentation." In addition to the results of the instruments, data were collected daily in the individual sessions with the participants and the computer. A cassette tape recorder equipped with an external microphone recorded each session.

The researcher also kept daily notes in a field notebook. In it student gestures and expressions were described and all of the commands that the participants typed into the computer were listed. Student drawings were filed in the notebook and the researcher sketched in the notebook the graphics produced by the participants on the monitor.

**Analysis of the Data**

Instruments.--The results of the pretest and posttest instruments (Piagetian tasks and student interview) were analyzed for evidence of change. The parent questionnaires were studied for behavioral changes at home related to the computer and student affect toward the computer. The
programming tasks were analyzed to determine the programming competence of the five-year-old participants.

Field notes.—The field notes included the dribble files of each student, descriptive comments about nonverbal behavior, and the transcriptions of the cassette tapes which recorded the daily sessions with the participants for the ten-week duration of the study. The tapes were transcribed in six months following the conclusion of the study. A set of headphones was plugged into the tape recorder to facilitate better understanding of the dialogue on the tape. A foot pedal was attached to the tape recorder for intermittent stops of the tape while typing the transcriptions. After the transcriptions were complete, several copies were made. The data were then sorted according to the categories of student behavior with the computer cited in the purposes of the study: (1) conceptual difficulties, (2) perceptual difficulties, (3) physical difficulties, (4) affective responses, and (5) thought processes. Each of these principal categories was organized into subtopics. The data were analyzed in each area and generalizations were written supported by specific instances from the transcriptions.

Case studies.—Individual case studies were written about each participant in the study. In the case studies student computer programs were examined, programming styles
were identified, and their attitudes toward working with
the computer were described.

Summary

Chapter III presented the methodology required to con-
duct this study. The procedures and conclusions of the
pilot study were discussed first and were essential to the
design and the execution of the present major study. A
detailed analysis of the data follows in Chapter IV.
CHAPTER BIBLIOGRAPHY


CHAPTER IV

ANALYSIS OF DATA AND FINDINGS

The problem of this study was to determine whether five-year-old children could learn to program a computer using the LOGO language and to describe and analyze their interactions with the computer. The purpose of Chapter IV is to analyze the data relating to this problem. There were three types of data gathered during the study: (1) student responses on the instruments, (2) field notes which were kept daily in a notebook and combined with the tape recorded transcriptions of the corresponding day, and (3) computer printouts of the graphics and procedures saved by students on their floppy disks.

The analysis of the data is also divided into three sections. In the first section the results of the instruments are examined in the following order: screening test, Piagetian tasks, student interview, parent questionnaire, and programming tests. A discussion of the results as they relate to the problem of the study follows.

The second part of Chapter IV contains the analysis of the field notes and transcriptions. The data were analyzed according to five categories of student behavior exhibited while working with the computer: (1) conceptual difficulties
(2) perceptual difficulties, (3) physical difficulties, (4) affective responses, and (5) thought processes. The data from the various categories are discussed in relation to the central question of programming competence.

The third section of the chapter contains the analysis of the students' computer programs. The printouts of the programs were analyzed along with the transcriptions of the sessions in which they were programmed. The data were analyzed according to three categories: (1) computer programs, (2) programming style, and (3) attitude toward programming. A discussion of the case studies follows.

Instruments

Screening Test

One section of the screening test was administered again at the conclusion of the study. This section of the test allowed students to demonstrate their understanding of number quantity. Students were asked to point to one of two numbers on a card in response to the question, "Which is more?" Students were not required to answer each of the three items correctly before they were eligible to participate in the study. (See Appendix B.)

Three students made one error each on the number quantity section of the screening pretest. When the section was given again as a posttest, one of the students repeated her
initial error. The other two students, both of whom were boys, responded correctly on the posttest. This result may indicate that LOGO computer programming aided the understanding of relative number quantity of some of the participants of the study.

**Piagetian Tasks**

Nine Piagetian tasks were administered at the beginning of the study (prior to computer use) and at the end of the study. The responses of the students to the tasks provided information about their relative cognitive development. Sometimes there was a discrepancy between the modal age at which a particular level of conceptual understanding is attained and the chronological age of the student.

Six of the tasks involved the concept of conservation. "Piaget defines conservation as the idea that some properties of a substance, such as amount or weight, remain the same, are invariant, are conserved, despite certain transformations" (Formanek & Gurian, 1976, p. 5).

**Charting Intellectual Development: A Practical Guide to Piagetian Tasks** (Formanek & Gurian, 1976) was used to interpret student responses on the tasks. This book conceptualizes the process of acquiring conservation as being comprised of three developmental stages. Stage I corresponds to the inability of the child to conserve on a task. Such a response could be labeled preoperational according to
Piaget's theory. Stage II is identified as a transitional phase of cognitive development in which the child gives inconsistent responses due to the influence of visual appearances during the transformations. The hallmark of Stage III is the attainment of conservation, a sign that the child has developed the mental structures necessary for concrete operations. Reference to these three stages is made in the analysis of the student responses on the Piagetian tasks.

Formanek and Gurian (1976) also include characteristic ages at which children function in each stage. Although Piaget's theory of intellectual development is not based on age norms, the ages cited in the guide are included as a basis for comparison in the analysis below. Most children attain the third stage of concrete operations between the ages of six and eleven.

The following section provides a brief description of each of the Piagetian tasks administered during the study. The conservation tasks are analyzed first. Then the classification, seriation, and shape drawing tasks are examined. The responses of the participants on the pretest and posttest of each task are analyzed and compared.

**Conservation of substance.**—The purpose of this task was to determine whether the participants were able to understand that the amount of a substance remained constant
in spite of various changes in its physical appearance. Two balls of clay equal in size were placed in front of the student. The student was asked whether the balls had the same amount of clay. After the student agreed they were equal, one of the balls was flattened. The student was then asked which had more clay—the ball or the pancake. Or did they have the same amount of clay? Two other transformations of one of the balls of clay took place. It was rolled into a sausage shape, and later it was broken into pieces. The student was asked to decide whether the ball or the clay in its new form had more substance, or whether they had the same amount following each transformation.

Students gave the same types of answers on the posttest as they did on the pretest. If they were unable to conserve substance on the pretest, they were nonconservers on the posttest as well. Two boys and one girl were unable to conserve substance on the three transformation tasks which is typical of five-year-old children. They focused on the physical differences between the ball and the transformation such as unequal length or width.

Two students (both girls) were able to conserve substance on the pretest and posttest. One of them reasoned that the two were the same despite the physical transformations because, "You haven't took any away." Children usually acquire the notion of the conservation of substance at about eight years of age.
Conservation of number.—The purpose of the task was to determine whether the participants could conserve the number equality of two sets despite transformations in the arrangement of one set along length or density dimensions. The materials used in this task were red and black plastic chips. The following three transformations were presented to the students.

A. Eight black chips were placed in a row and the child was given a pile of ten red chips. The child was asked to fix the red chips so there would be as many as the black chips.

B. The red chips were placed in a row so that the sides of the chips were touching. The black chips were put in a circular configuration with the sides touching.

C. The red chips remained in the same row. The black chips were spread out in a long row with space in between the chips.

After transformations B and C, the child was asked which had more chips, the red or the black, or were they the same?

The analysis of the students' responses revealed that all of the participants were able to make the sets equal in number in Transformation A. They accomplished finding an equal number by lining up the two sets using one-to-one correspondence or by counting the number of chips in each set and setting aside two red chips.
Students responded consistently on the other two transformations. If a student was a nonconserver on the pretest of Transformation B, he was also unable to conserve on the pretest of Transformation C. Two girls were able to conserve number on the pretest and posttest on both Transformations B and C. This is a Stage III response typical of a child who is seven or eight years old.

Two students exhibited a shift in their responses on the pretest and posttest of the conservation of number task. One girl demonstrated an inability to conserve on the pretest of Transformations B and C (a Stage I response of under five years) yet, by the time she took the posttest, she had acquired the concept of number identity. She said that both sets were the same because they "both had eight" (stage III response). Another boy shifted from a Stage II transitional response on the pretest to a Stage III response on the posttest. He, too, had acquired the ability to conserve number equality by the end of the study.

The remaining student's pretest and posttest responses indicated that he was in transition between nonconservation and conservation (Stage II). He was confused by the change in the appearance of the chips, yet he acknowledged that both sets contained eight chips. This kind of response is typical of a child five or six years old.
Conservation of liquids.—The participants had to decide whether the amount of liquid remained the same as it was poured into glasses of varying sizes in this conservation task. The materials used were two standard size drinking glasses, a pilsner glass, three small juice glasses, and two equal amounts of water. Once the students had agreed that there was the same amount of water in both of the drinking glasses, one of the glasses was emptied into the pilsner glass. It was compared to the filled drinking glass. The water from the pilsner glass was returned to its former drinking glass. Then it was poured into three juice glasses. Finally, the water from the juice glasses was returned to the original drinking glass. Following each transformation (pouring of the water), the students were asked, "Which has more water or are they the same?"

Student responses on the pretest and the posttest were the same. There was no change in the students' developmental understanding of the concept of conservation of liquids during the ten-week study.

Three of the students (two boys and one girl) were unable to conserve on this task. When asked to compare the quantity of liquid in the drinking glass to that in the pilsner glass, a typical student response was that the pilsner glass had more water because "it's more taller." They focused on the difference in height rather than realizing that the change in the width of the glass compensated
for the height of the liquid in the glass. This kind of response is classified as a Stage I response, usually given by children who are four to six years old.

The other two participants (both of whom were girls) were able to conserve the amount of liquid on the pretest and posttest. They were able to take the two dimensions of height and weight into account. When the examiner questioned one student's response by saying, "They're the same? This one looks taller." She replied, "Yeah. But it's more skinnier." The other girl justified her answer by saying, "Cause there's the same in that one as there is in this one, and then you pour it into here, so I know it's the same." She was using the "identity operation: when the child realizes that nothing has been added or removed, [she] knows that the number [amount] remains the same" (Ault, 1977, p. 63). "They're the same," the student added, "cause you haven't let me drink any." These responses indicating conservation are typical Stage III responses. This developmental level is not generally attained until the age of eight years. Therefore, these students gave responses which were developmentally advanced for their chronological age.

Conservation of length.—Students were presented with two identical pens or two unsharpened pencils and asked to compare the length of one set. (If the child thought the length of the first set of objects was unequal, the second
set was offered for comparison.) Once equality of length was established among the two objects, their relative positions were changed. The objects were aligned and one was moved forward slightly on the table. Students were asked, "Which is longer or are they the same length?" Then the position of pens was reversed and the other pen was pushed forward. The same question was repeated. The purpose of the task was to determine whether students could conserve the equality of length throughout the visual transformations.

Students' responses were consistent on the pretest and posttest of this task. Two boys and one girl were unable to conserve the length of the objects. They focused on the one which projected forward farther and said it was longer, a Stage I response.

The other two students (both girls) realized that no change had taken place in the length of the pencils in spite of the visual differences. One girl used the identity operation to justify her Stage III response: "They're the same because you haven't cut them."

**Conservation of distance.**—Students were presented with two paper dolls of a boy and a girl and two dividers. The dolls were placed about twenty inches apart on the table. Students were asked whether the dolls were "near together or far apart." A paper replica of a fence was placed halfway between the dolls. Students were asked again
whether the dolls were near together or far apart. Another fence was put between the two figures in the same position. The replacement fence had spaces between the bars. Children were then asked whether the girl and boy were near together or far apart.

The purpose of this task was to determine whether students could conserve the distance between two objects whose relative position did not change. For children in Stage I, under five years, "the introduction of a screen [fence] changes the distance relationship between the two toys. The child cannot deal with one distance broken into two parts. He now considers a part of the whole with which he began and finds that the distance is less or more" (Formanek & Gurian, 1976, p. 33). None of the participants in the study gave Stage I responses.

Two students appeared to be in the transitional Stage II, typical of children five to seven years old. In this stage, children believe that a difference in space occurs because the fence takes up space. One girl responded that the boy and the girl were far apart at the beginning, but were near when the fence was interposed. One of the boys gave a Stage II response on the pretest and a Stage III response on the posttest.

The other three students' responses on the pretest and the posttest could be classified as Stage III responses, characteristically given by children seven years and older.
They were able to conserve distance. Their answers were consistent; the introduction of the fence did not distract them. One girl explained that the girl and the boy were the same distance apart, "because you haven't moved them."

After analyzing the responses of the students, the effectiveness of the task must be questioned. The consistent Stage III responses may have referred to the child's concept of near and far rather than to the ability of the child to see two objects equidistant in space regardless of intervening barriers. When the examiner questioned one student why she was saying that the figures were far apart, the girl replied, "Cause one's over here and one's over here. If they were near, they'd have to be like this" (she moved the figures next to each other). The same type of justification was given by three other students.

Conservation of area.—This task was designed to determine whether children could conserve the area of two equal spaces despite visual differences in the arrangement of equal objects on the spaces. The materials for the task included two identical sheets of green construction paper, two small paper cows, and eight small blocks of equal size.

Students were asked to pretend that the adjoining construction paper areas were fields of grass. A paper cow was placed in each field. The question was asked: "Which cow has more grass to eat? Or do they have the same amount?"
A barn was placed on each field. The same question was repeated. The examiner continued to add up to four barns on each field of paper and repeated the question with each new addition. On one field, the barns were placed in a row. On the other field, the barns were scattered across the area. Although each field contained a cow and an equal number of barns, the configurations of the barns and the remaining space were visually different.

Student responses to the task spanned the range of developmental levels. One boy was unable to conserve area on both the pretest and posttest. He insisted that there was more grass on the field in which the barns were spread apart. This is a typical Stage I response, usually given by children who are under five years old.

The other boy's inconsistent answers on the pretest indicated that he was in the transitional phase of Stage II. By the time he took the posttest, he had developed the ability to conserve area. His responses to the task had changed to acknowledge that each cow had the "same amount of grass," a Stage III response typical of children seven years and older.

The three girls were able to conserve area on the pretest and posttest. They counted the barns on each field as their strategy to justify that the cows had the same amount of grass to eat. One girl said they were the same because "these pieces of paper are the same size" and because "they both have four [barns]," a strong Stage III response.
Classification of geometric objects.—The purpose of this task was to determine whether the participants could spontaneously classify geometric objects. The materials included paper circles, squares, and triangles of five different colors. The shapes were spread out randomly. Students were instructed to "put together the ones that go together."

Students demonstrated various developmental responses to the classification task. One boy appeared to be in transition from Stage I (under five years) to Stage II (five to seven years). He first classified the paper shapes by colors and then proceeded to arrange the shapes into configurations which had meaning for him: a person, a cat, and a "rocket car." These representations of objects are called graphic collections and are typical of children up to five years of age. The classification of shapes according to either color or shape is a typical Stage II response. This boy sorted the shapes by color and made graphic collections during the classification pretest and posttest.

A girl sorted by shapes only on the pretest and posttest. She exhibited a Stage II response characteristic of children five to seven years old.

The other three participants (two girls and a boy) seemed to be in transition from Stage II to Stage III by the time they took the posttest. These three students were all able to classify by shapes and by colors on both tests.
However, on the posttest, they included subclasses of shapes and colors in their classification schemes. This indicates that the students were making progress toward reaching Stage III, class inclusion, which is generally associated with children eight years old.

Although the responses of two students did not change over time, three students exhibited more developmentally mature responses on the classification posttest.

**Seriation.**—The purpose of the seriation task was to determine whether students could rank order objects according to length. It was speculated that there might exist a correlation between perceiving differences in lengths and the ability to create satisfactory graphics with the Turtle, both of which require good visual perception and organization skills.

The materials for the seriation task included thirteen blue dowel rods of varying lengths and equal diameter. The rods were presented in a random array. The participants were asked to put the sticks in order. If a student did not know how to respond to that directive, he or she was told to arrange the sticks going from the "biggest to the littlest."

The three female participants understood the meaning of "in order" and quickly put the rods in the correct order. It was evident that they were sorting systematically
according to length by surveying the entire set before they selected the appropriate rod.

On the pretest, one boy needed the phrase "in order" explained to him. However, once he understood the directive, he completed the task correctly. He did not require any assistance on the seriation posttest.

The preceding four participants were functioning at Stage III on seriation. Children are usually seven to eight years old before they order elements in a series successfully.

The remaining boy required additional explanations and prompts during the pretest and posttest of the seriation task. His responses to both tests are representative of Stage I (under five years) in which the rods are divided into two groups, such as small and large. This child did not make a separation between the groups; rather he placed the rods in a line and put several large rods on the left and a group of smaller rods on the right. There were eight rods of contrasting sizes in an alternating pattern in the line. He may have arranged them according to a representational scheme. When the examiner inquired, "What kind of order are you putting them in?," he replied, "like a fence." He did not align the rods with the edge of the table nor with each other as the other students did. His line of rods was straight, but the ends of the rods were uneven. Thus his response to the seriation task was developmentally immature
for his chronological age and significantly different from the responses made by the other participants on this task.

Drawing shapes with topological and Euclidean properties. Participants were asked to look at a set of cards with various shapes and copy each card on paper using a crayon. This Piagetian task was designed to assess the child's concept of space. Formanek and Gurian (1976) distinguish between two types of spatial discriminations: "Topological deals with the open and closed figures such as a simple closed curve. Euclidean deals with angularity, parallelism, and distance" (p. 85). Developmentally, children acquire topological concepts first, usually by the age of four. Then they develop the concepts of projective space, "where objects are viewed, not in isolation, but with regard to other objects. This is followed by concepts of Euclidean space, which deals with geometrical figures, angles, etc." (Formanek & Gurian, 1976, p. 85).

This task was administered to determine the conceptual levels of spatial discrimination displayed by the participants. In the study, LOGO graphics required participants to utilize the three types of spatial concepts: topological, projective, and Euclidean.

An analysis of the student drawings of the model shapes revealed few differences between the pretest and posttest results. Student responses to this task were representative
of the developmental range of children's conceptual understanding of space.

One boy appeared to be in transition from Stage I (four years) to Stage II (five to seven years). His triangles and squares were indistinguishable at times, as were his circles and ellipses. His drawings displayed difficulty with angularity and projective space.

Three students (two girls and a boy) exhibited good perception of spatial relations. Their drawings could be classified in Stage II (from five to seven years), which corresponds to their chronological age. The responses of these three students displayed some difficulty with the Euclidean properties of angularity and parallelism. The students also made spatial discrimination errors on the composite figures, such as the triangle within a circle.

The remaining student, a girl, had excellent spatial discrimination. Her nearly perfect responses were characteristic of Stage III, responses typical of children aged seven years and older. Her copying was very accurate on the pretest. She drew pictures of the figures well again on the posttest; however, this time she added original details to the initial figures. The details transformed the abstract geometric symbols to representations of familiar objects. She changed the circle into a face by adding features and she made the rectangle into a house. These unexpected additional details made the drawings meaningful to her.
This behavior resembles the first stage of classificatory grouping, graphic collections, in which the geometric shapes are combined to represent objects with which the child is familiar.

Summary.—Student responses were generally stable on the pretest and posttest of each Piagetian task. Thirty-six responses out of a total of forty-five remained the same on both tests. Of the nine changes that did occur from the pretest to the posttest, four students made developmental advances on the classification task and two students made progress in the conservation of number. The other three changes were made by the youngest boy in the study on the conservation of distance and area tasks and on the shape drawing task. In all, the two boys accounted for two-thirds of the total number of changes from the pretest to the posttest and the girls made one-third of the changes. This disproportionate ratio can be attributed to the fact that two of the girls gave Stage III responses to all six of the conservation tasks on the pretest. Thus they had already achieved conservation and their responses remained the same on the posttest.

Table II compares the performance of the individual students on the Piagetian tasks. The responses of the girls and the boys are grouped together for comparison.
TABLE II
COMPARISON OF PERFORMANCE ON THE PIAGETIAN TASKS*

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Girls</th>
<th></th>
<th>Boys</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Alexandra I</td>
<td>Andrea III</td>
<td>Danielle III</td>
<td>Billy I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Jonathan III</td>
<td></td>
</tr>
<tr>
<td>Substance</td>
<td>I</td>
<td>III</td>
<td>III</td>
<td>I</td>
</tr>
<tr>
<td>Number</td>
<td>I-III</td>
<td>III</td>
<td>III</td>
<td>II</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>II-III</td>
<td></td>
</tr>
<tr>
<td>Liquids</td>
<td>I</td>
<td>III</td>
<td>III</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Length</td>
<td>I</td>
<td>III</td>
<td>III</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Distance</td>
<td>II</td>
<td>III</td>
<td>III</td>
<td>II-III</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>III</td>
<td></td>
</tr>
<tr>
<td>Area</td>
<td>III</td>
<td>III</td>
<td>III</td>
<td>II-III</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>Classification</td>
<td>II-III</td>
<td>II</td>
<td>II-III</td>
<td>II-III</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>I-III</td>
<td></td>
</tr>
<tr>
<td>Seriation</td>
<td>III</td>
<td>III</td>
<td>III</td>
<td>I</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>III</td>
<td></td>
</tr>
<tr>
<td>Drawing</td>
<td>III</td>
<td>II</td>
<td>II</td>
<td>I-II</td>
</tr>
</tbody>
</table>

*Roman numerals refer to stages of development cited in Formanek & Gurian (1976). Two numerals indicate change in the responses given on the pretest and posttest.

The two boys had five Stage I responses on the pretest and four Stage I responses on the posttest. One girl's performance resembled that of the boys. Alexandra had four Stage I responses on the pretest and three on the posttest. However, her performance on the last three tasks was more advanced than the boys' responses on those tasks.

The responses of Andrea and Danielle were very similar. Most of their responses were classified as Stage III. Both
girls were in the stage of concrete operations on the conservation tasks.

The level of performance on the Piagetian tasks was unrelated to the chronological age of the students. Jonathan and Andrea, the two oldest students at five years and eleven months when the study began, gave disparate responses to the tasks. Two girls who were nearly the same age (5.8 and 5.9 years), Alexandra and Danielle, also gave differing responses to the tasks.

Table III provides a tally of the number of student responses in each of the developmental stages of the acquisition of conservation. The right half of the table converts the numbers into percentages of the total number of student responses.

**TABLE III**

**PRETEST AND POSTTEST RESPONSES CLASSIFIED ACCORDING TO STAGES**

<table>
<thead>
<tr>
<th>Stage</th>
<th>Number of Student Responses</th>
<th>Per Cent of Total Responses</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pretest</td>
<td>Posttest</td>
</tr>
<tr>
<td>Stage I</td>
<td>14</td>
<td>11</td>
</tr>
<tr>
<td>Stage II</td>
<td>12</td>
<td>8</td>
</tr>
<tr>
<td>Stage III</td>
<td>19</td>
<td>26</td>
</tr>
<tr>
<td>Total</td>
<td>45</td>
<td>45</td>
</tr>
</tbody>
</table>
About one-third of the students gave preoperational responses on the pretest. Only three students changed their responses on the posttest. Numerical results were similar for Stage II responses. Therefore no great change in developmental level can be attributed to the computer programming intervention. This is probably due to the maturational nature of the Piagetian tasks. The ten-week period of the study was a relatively brief time span in which to observe developmental changes.

Stage II had the fewest number of student responses. This was surprising since children who are five and six years old usually give Stage II responses according to Formanek and Gurian (1976). However, the age at which children achieve the various levels of development changes depending on their cultural environment and experiences. The interpretive manual used did not specify the population or culture of the children used to norm the tasks.

There were more Stage III responses given than any other category. Typically children are seven or eight years old before they perform at this level. It is unusual that between 42 and 58 per cent of the students in the study responded at this level of concrete operations while they were five years old. The student sample appears to be atypical in relation to their relatively advanced level of performance on the Piagetian tasks. It could be concluded that this homogeneous group of children (in terms of race
and socioeconomic status) was unusually mature in terms of intellectual development from the beginning of the study. Nevertheless, there was some diversity among student responses. Students' responses were categorized into two or three different stages on each task. Overall, student responses spanned the range of developmental stages, although there was an unexpectedly high percentage of Stage III responses.

**Student Interview**

An informal interview was conducted with each of the five participants at the beginning of the study (prior to working with the computer) and again at the conclusion of the study. The purpose of the interview was to assess student attitudes toward working with the computer and to gather data about their perceptions of how the computer worked and of their role in its functioning.

The interview questions are listed and analyzed below. Some of the items were utilized only on the pretest or on the posttest; they are designated accordingly. The other questions were asked on the pretest and on the posttest.

1. What do you like to do at school? (If they say "play," ask: What do you like to play with at school?)

Students responded with various answers. Several said they liked to "play outside." All of their responses related to class activities. On the posttest, none of the students
mentioned the computer. This may be due to the fact that the computer was located in a separate room which was distant from the kindergarten classrooms. Thus they may not have thought of their work with the computer as a school activity.

2. If you could have anything at all, what would you want?

Answers on the pretest included lists of toys. On the posttest, two students changed their answers to include references to the computer. One student said she wanted a "typewriter" (like the keyboard of the computer), and another student said he wanted a computer.

3. (pointing to the computer) What is this called?

Pretest responses included "a typewriter and a television" and "a typewriter—I've never seen one like that." The other students said they did not know what to call it. On the posttest, four students called it "a computer." One student said that she didn't know the keyboard was part of the computer. In addition to correctly naming the computer, two students identified the monitor, the disk drive, and the floppy disk.

4. Do you have a computer at home? (pretest)

None of the participants had a computer at home.

5. Have you used a computer at school before? (pretest)

None of the students had used the computer at school before the study began.

6. Tell me how it works. (pretest) Tell me how the Apple II Plus computer works. (posttest)
On the pretest, two students thought the computer would work like a typewriter. They referred to inserting paper and typing. One of them said "it would show you up there (on the screen) what you writed [sic]." The other students did not know how it would work.

All of the participants referred to turning on the switch or the monitor in the posttest responses. "It makes it work from the electricity." Four of the five students mentioned the floppy disk in relation to the operation of the computer. Their explanations varied in detail from "by floppy disks" to "You have to put the floppy disk in the disk drive."

Four students included typing as an operation needed to make the computer work. Although two of them could not recall the word typing, they clarified their meaning by pointing to the keys or making typing motions. Their descriptions included "pushing the dials (keys) on it" and "writing with the keys--these little square things." No references were made to inserting paper in the computer on the posttests. Evidently that misconception was corrected as the students worked with the computer.

7. What do you do when you work with it?

On the pretest most of the respondents thought that the computer was a typewriter or a kind of writing machine ("because it's got letters"). The posttest responses indicated a change had occurred during the study in the
students' perceptions of the function of the computer. Participants viewed the computer as a machine that would draw pictures when given commands. Typing was mentioned by some students in their descriptions of what people do when they work with it, but it was related to the act of giving commands for the purpose of drawing pictures. One participant expressed an atypical posttest response: "Play DRAW and all those other games." He viewed the computer as a kind of videogame machine. He enjoyed playing the preskill games in which he manipulated the Turtle with the keys F, B, R, and L.

8. How do the pictures get on the computer's TV screen? Students responded in a variety of ways to this question. Most answers made reference to television operating procedures or parts such as "push a button, turn the channels, and by the wires." On the posttest, four participants said that the pictures got on the computer's screen through the process of typing in commands. A representative answer was "by giving it commands and drawing them with the Turtle." Thus students changed their perspective during the study from manipulating the parts of the "television" set to understanding the cause-effect relationship of typing in commands and creating pictures on the screen.

9. What do you think the computer is? Pretest responses referred to things which physically resembled the visible components of the computer:
"typewriter, or a television." More generic responses included "a machine" and "a thing that's electric."

In general, posttest responses expanded the original notions about the computer. "It's like a TV but it has letters and games on it." "It's a computer—it's not a toy to play with. Something you work on." Only one student's answer did not change over time. She referred to the computer as being a typewriter on the pretest and on the posttest.

10. How do you think the computer thinks?

Most students did not know how to respond to this question. One girl (who said she had seen a computer on the television news) believed the computer thought "with the stuff in there" and pointed to the disk drive.

The posttest responses differed from those given on the pretest with one exception. One boy responded "I don't know" on both tests. On the posttest, most students indicated that they told the computer how to think by typing in commands. One girl said the computer thought "from his brain in here" (referring to the inside of the computer) with the "little rectangles" (cards of silicon chips).

11. What do you do when you think?

There was little variation among the pretest and posttest responses to this question. Answers were very limited, such as "think about things, think about doing something, and think of something nice." This question was not
appropriate to the developmental level of the participants. They found it difficult to talk about thinking as an intellectual process; their answers were generally related to outcomes. For example, one boy said, "I think of doing numbers." Then he counted aloud from one to fifty.

12. Pretend the computer can understand what you say to it. What would you tell it?

Two students said they would write with it in their responses to the pretest. Other pretest responses were diverse, such as "something nice" and "What are you doing, computer?" Answers on the posttest reflected a shift in perspective. Most students said they would tell the computer to do the things (commands) they told it. Students realized they had the power to make the computer do what they wanted by the time they responded to the posttest interview. One girl said, "I would tell it: Make something that I like."

13. How do you feel when you work with the computer? (posttest)

All of the responses were positive. Two students had difficulty recalling words which expressed feelings. The examiner then provided a range of examples of feelings. Several students said they felt happy when they worked with the computer. One boy remarked, "Sometimes you learn how to know things. I feel like it's fun."
14. What do you like about it? What is fun about it? (posttest)

The three female students replied that they liked typing and drawing with the computer. The perception of what was fun about the computer was very different for one boy. He liked the preskill game G, which he called "the Turtle in the Box: F, B, R, and L." When asked whether he liked making pictures with the computer, he responded, "I think that's hard to do. But I don't feel like it will be very much fun at all."

The other boy liked the mechanical operation of the computer: inserting the disk in the drive and the inside of the computer ("wires, little steel things that the wires go through and all that stuff"). He knew that he was not supposed to touch the exposed parts of the floppy disk. The fact that he was forbidden to touch them intrigued him. He conjectured, "Maybe that's sharp (the circular edge of the floppy disk) and you can cut your fingers on it, if you press down real hard... The thing in there (inside the disk drive) knows what it feels like."

15. What do you not like about the computer? What do you think is hard? (posttest)

Some students expressed frustration with correcting mistakes ("You have to waste your time fixing it"). They also did not like trying to cope with mechanical difficulties,
such as the time the monitor broke down. Two students said that there was nothing they did not like about the computer. All students had responses for the question "What do you think is hard?" Two students thought making a triangle was hard. Other responses included "typing in the stuff" and "things I don't know about the computer."

The student interviews provided a great deal of information about student perceptions of computer functioning and how they felt about working with it. Student perceptions of how the computer worked were definitely altered by the study. Early notions of the computer as a typewriter or a television had changed by the conclusion of the study. The computer was later perceived as a machine that had to be switched on, the floppy disk inserted into the disk drive, and typing had to take place in order for it to function. At the end of the study, the participants recognized that typing in the commands created the pictures on the monitor. They understood that they had the power to make the computer "do the things I tell it."

Posttest responses concerning the way students felt when they were working with the computer were all positive. The participants enjoyed typing and making pictures with the computer. Some of them were fascinated with how it worked. They wondered how the disk drive translated the information from the floppy disk and wondered about the impressive array of silicon chip cards inside the computer.
Students identified tasks which frustrated them (corrections and mechanical problems) as things they did not like about the computer. They thought making a triangle was hard and typing in long programs was hard.

In general, the participants liked working with the computer and felt successful about "making things" with it. They did experience some frustrations in working with it, however, and identified the tasks which were difficult for them.

**Parent Questionnaire**

A one-page questionnaire was sent home with the participants for their parents to answer. The purpose of the questionnaire was to obtain information about the feelings or comments made at home by the participants concerning their work with the computer. The questionnaire also asked parents to write down any instances of behavioral changes related to computer programming that they had noticed in their child's play, drawing, or writing activities. (See Appendix D.) All of the questionnaires were completed and returned. Some parents answered in greater detail than others.

Four mothers and two fathers observed their children's work with the computer during the study. Through these observations the parents became familiar with Turtle graphics, LOGO commands, and the procedures and programs their children had executed. Thus they were able to give informed
responses to the questions concerning the influence of computer programming on their child's behavior at home.

All of the participants talked to their parents about working with the computer. Most parents reported that their child talked about it often. Some of the participants described the programming projects to their parents and explained what they accomplished each day. One mother said,

She has told us about the Turtle and what she has to do to make the Turtle move. She told us about HT—using the abbreviations seems to be a neat thing for her. She's told us about making letters, a house, and a garbage can. She seemed to be especially pleased at finishing each of these projects. She also explained how she made some of them, i.e., 70 up, turn right or left, go x number, etc.

Some of the participants' comments to their parents were more general. They stated that they "had fun" with the computer, or they drew pictures with it.

Only one parent noticed any behavioral changes at home which were related to computer programming. She said that her child "related different words or concepts from the computer to other things." The child remarked about spatial relationships at home, like "going backwards," and number quantity. The mother stated, "She also asked us what number was larger than another. She was always doing little number games and asking to spell things." One of the kindergarten teachers said that at the art center, the participants drew pictures of the Turtle and particular letters or shapes they were programming.
The fourth item on the questionnaire asked parents to circle the emotions their child had demonstrated or expressed at home concerning their work with the computer. Every parent circled the descriptor "happiness." The word "competence" was marked by four of the five parents responding. (The parent who did not circle this descriptor remembered that her child said that "making the Y was hard.") Other feelings expressed at home by the children about their experiences with the computer included "enthusiasm" and "self-confidence." None of the descriptors referring to negative feelings (such as "frustration" or "anxiety") were circled by the parents.

According to the parents' written responses on the questionnaire, each of the participants had a positive experience with the computer. Two parents stated that their children eagerly looked forward to working with the computer each day. The participants talked to their parents often about the computer and their programming experiences. One parent noticed that his child "spoke more of the computer work than other school activities."

Most parents did not notice changes in the participants' play, drawing, or writing activities at home which related to their work with the computer. However, one kindergarten teacher who responded to the questionnaire said that she noticed work related to programming when the participants were at the art center. Proximity in terms of distance and
time may have influenced the production of such related work. The participants may have produced drawings about their computer projects only at school because they were motivated to express what they had just worked out on the computer and art materials were always available in class.

**Programming Tests**

Two types of programming tests were administered at the conclusion of the study: dot tasks and shape procedures. (See Appendix E.) The computer was programmed to create configurations of four dots on the monitor. (See Appendix F for programming commands.) There were four sets of the dot tasks. The students were asked to make the Turtle connect the dots. The dot tasks were designed to allow students to demonstrate their knowledge of number quantity and spatial relationships. The answers for the correct resolution of the dot tasks are provided in Appendix G. Responses which were within a range of four steps from the exact position were considered acceptable.

Student participants were also asked to write or dictate procedures for three shapes. The purpose of this test was to determine whether students could write workable procedures without assistance for a square, a rectangle, and a triangle. Writing shape procedures required students to demonstrate their understanding of the structure of a procedure and the sequential order of commands.
Students were asked to complete the test on paper first, without using the computer. The shape was already drawn on the paper and they were asked to indicate the corner where they wanted the Turtle to begin drawing an identical shape. After they had finished giving the commands for the procedure, they were asked to read them again. Students were given an opportunity to make changes in their procedure. Then the procedure was typed into the computer to verify that it worked. If the resulting graphics were unsatisfactory, students were encouraged to review the procedure, find the errors, and correct them. The revised procedure was checked on the computer. This process was repeated until the shape was successfully produced on the monitor or until the participant decided to stop working on the problem. Students were allowed to work on a procedure for one session only. No "second chances" were given in the event of unsuccessful attempts. Some students were able to complete two procedures in one session.

Dot tasks.—The first three dot tasks were completed successfully by all of the participants except one. Billy overestimated the last move of the first dot task by ten steps. He was unaware of his error because the line covered up the dot on the monitor.

The fourth task was more difficult because the dots were positioned at varying distances and the last move required
a forty-five degree angle input. Alexandra moved the Turtle ten steps too far on the first phase of the fourth task. She did not notice her error. Otherwise, all students made acceptable responses to the first three parts of the fourth dot task.

The final move of the fourth task was difficult for all of the students. Only one, Danielle, succeeded in correctly solving the problem on her fourth trial. Finding the correct angle to the last dot was the major problem. She tried the following angles in succession: RT 70, RT 50, RT 40, and RT 45. (Most students used a PENERASE, BACK, PENDOWN strategy to restore the Turtle to its former position after errors were made. Then they would try a new input.)

Alexandra also tried to solve the problem four times. With each trial she came closer to solving the problem successfully. The angle inputs she tried were: RT 90, RT 60, RT 7, and RT 50.

Billy attempted the move twice with RT 90 and then RT 80. He became frustrated and said, "I just can't understand how to get it up here. This is hard to do."

Jonathan approached the problem differently. Instead of trying to turn the Turtle to the correct angle all at once, he gave a few commands moving the Turtle to the right and then a few commands moving the Turtle forward. He never used an input larger than nine. This succession of small moves to the right and forward ended in getting the Turtle
to the fourth dot, but the line was not straight and the angle was not forty-five degrees. The totals of his inputs were: RT 36, FD 36, RT 25, FD 6, RT 26, and FD 7.

Andrea reached the dot with a straight diagonal line, but without understanding how she did it. She randomly selected the angle input of RT 764. This had the effect of causing the Turtle to turn two complete circles and approximately forty-five degrees. Thus her lucky guess achieved the desired result, but her response could not be considered a correct one.

**Shape procedures.**—Two students, Danielle and Jonathan, wrote acceptable procedures for all three shapes. Alexandra and Andrea succeeded in writing procedures for the first two shapes, but failed to write a workable procedure for a triangle. Billy dictated a correct procedure for a rectangle, but was not successful with the other two shape procedures.

Writing procedures for a square and rectangle was relatively easy for most students. All of the participants were able to write correct procedures for a square and a rectangle except Billy. Writing a workable procedure for a triangle was much more difficult for all of them. The two students who succeeded in making a triangle worked out the angles on the monitor. They continued to experiment and watch the turning of the Turtle until they achieved the desired angle. The other three students wrote RT 90 when
they wanted the Turtle to turn in their triangle procedures. (An angle of one-hundred-twenty degrees is required for an equilateral triangle.) They were surprised when it did not make a triangle, but did not attempt to substitute other inputs for the angle when they were encouraged to revise their procedures. In fact, they did not make any revisions at all.

These students may have believed that an input of ninety was the only way to make the Turtle turn. As Billy was working on the procedure for the triangle he said, "We have to turn it by pushing RT 90." Another explanation for their failure to try other inputs may be due to a perceptual problem. These students may not have noticed the difference in angle degree between a square and a triangle. The following excerpt from the transcription of the tape recorded conversation between Andrea and the researcher indicates a perceptual problem. Andrea had just typed her triangle procedure into the computer and noticed a major difference between the graphics on the monitor and the drawing of the triangle.

R Can you think of some commands that you'd like to change to make a triangle?
AN I don't know.
R Do you know what was wrong with it?
AN (shakes head no)
R What kind of an angle does ninety make, Andrea?
AN A sharp turn.
R Just like in a box or square. Look at this triangle. Does it have sharp turns in it?
AN (nods yes)
Okay. It has . . .
Three of them.
Three turns. But do these turns look like the same kind of turns that are in a box?
To me they do.
Do you have any ideas for wanting to change it?
(shakes head no)
Perhaps Andrea could not detect the greater angle input required for a triangle due to her age and stage of perceptual development.
Danielle, however, indicated that she noticed the angle difference needed for a triangle. When she started working on the third shape procedure, she remarked: "Now we have to get a bigger number [than ninety] to make it go like that [slanted]. I wish we could look up how on the computer, how we made the thing [slanted roof] of the house." Nevertheless, for the second turn of the Turtle, she tried RT 90.

Ninety, do you think?
That's your decision. We can try it.
(whispers) I think a bigger number. I need to try some stuff [on the computer]. I need to try some suggestions.
Do you want to try typing in what you've got so far? And then decide on your last RT number [input]?
Yeah. (types in program through RT 90) Oh-oh. Erase ninety (marks through RT 90 on paper).
What do you want to try instead of ninety up here?
(write 110) We'll try that. And if it doesn't work, we'll try 120.

Danielle differed from the other three students in that when she discovered RT 90 would not produce the correct angle of turning, she continued to try larger number inputs.
Jonathan also tried turning the Turtle with RT 89 in his triangle procedure. (Occasionally he used RT 89 instead of RT 90 in making a square.) When he revised his procedure using the monitor, he added a series of RT commands following RT 89 until the Turtle pointed to the desired angle. Both Danielle and Jonathan exhibited a great deal of task perseverance in varying the angle inputs until a satisfactory triangle was created.

Their triangles were not perfect, but they were acceptable. Jonathan's triangle was not quite closed. The lower left corner of Danielle's triangle was off the screen and one side extended past the upper intersection. She explained the aberration by saying: "Do you think that was a neat triangle? See, this is the triangle (tracing it with her finger on the monitor). And that's just the string to it (pointing to the extension)."

Students were given outline drawings of the shapes they were to program. They wrote their procedures on the same page, underneath the drawing. Most of the participants kept track of the Turtle's location by drawing an arrow or small triangle representing the Turtle on the shape after writing each command. Andrea was the exception in this regard. She made no marks on two of her shapes. All of the other students marked the various locations of the Turtle on all three shapes.
Students were familiar with the formal structure of a procedure. They had all named programs and used the words TO and END correctly prior to the programming tests. In writing the shape procedures, the girls used TO more consistently than the boys. Billy and Jonathan frequently forgot to write TO before the name of their shape procedure. Both boys and girls remembered to include END as the last command in their procedures.

Students were given the option of dictating their shape procedures or writing the commands themselves. Billy was the only participant who dictated all of his procedures. The other four students wanted to write their own procedures.

**Summary.**—The programming tests required students to respond in two different ways. The dot tasks were structured; students' answers were expected to conform to the defined limits. Responses were convergent; directional commands and number estimates were either accurate or inaccurate. Writing the shape procedures was a divergent method of testing programming skills. The students had to create the structure of the procedure as they named it, defined it, and ended it. Diversity was allowed in the numbers and turning commands used in the procedures. The evaluative criterion was that the procedure must result in a recognizable square, rectangle, or triangle on the monitor.
Some students were more successful on the programming tests than others. Table IV displays the individual results of the programming tests.

**TABLE IV**

**COMPARISON OF PERFORMANCE ON THE PROGRAMMING TESTS***

<table>
<thead>
<tr>
<th>Tests</th>
<th>Alexandra</th>
<th>Andrea</th>
<th>Billy</th>
<th>Danielle</th>
<th>Jonathan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dot Task 1</td>
<td>C</td>
<td>C</td>
<td>I</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Dot Task 2</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Dot Task 3</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Dot Task 4</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>C</td>
<td>I</td>
</tr>
<tr>
<td>Square</td>
<td>C</td>
<td>C</td>
<td>I</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Rectangle</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td>Triangle</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>C</td>
<td>C</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td>5:2</td>
<td>5:2</td>
<td>3:4</td>
<td>7:0</td>
<td>6:1</td>
</tr>
</tbody>
</table>

***"C"—correct resolution of problem, "I"—incorrect response.

**Total figures indicate the proportion of correct responses versus incorrect responses.

Danielle solved all of the tests correctly. Jonathan missed only one dot task. Alexandra and Andrea both failed to work out the forty-five degree angle in the fourth dot task and to write a correct triangle procedure. Billy solved more tests incorrectly than he did correctly.
Discussion of the Results of the Instruments

Two important questions remain: (1) Is student performance on the Piagetian tasks related to their performance on the programming tests? and (2) What conclusions can be drawn about the programming competence of the participants based on the results of the instruments? The following discussion addresses these concerns which the study sought to answer.

Michael Folk (1972) conducted a four-month study with eight fourth-grade children who were learning LOGO. He found a relationship between the developmental level of the subjects (as measured by Piagetian tasks) and their competence in using some concepts of computer programming.

The results of the present study are rather unclear regarding the relationship between the performance of the students on the Piagetian tasks and their performance on the programming tests. A comparison of Table II (Piagetian tasks) and Table IV (programming tests) indicates that although the girls generally performed better on the Piagetian tasks than the boys did, two of the girls were not as successful as Jonathan on the third shape procedure. Jonathan did well on most of the programming tests, yet he gave several preoperational (Stage I) posttest responses on the Piagetian tasks.

The responses of two students, Danielle and Billy, suggest a possible relationship between developmental level
and programming competence. Danielle was the most successful on the programming tests and she had attained the level of concrete operations on most of the Piagetian tasks. Billy's rank of performance on those two instruments was in direct contrast to Danielle's. His performance was the poorest on the programming tests and he gave more Stage I responses than any other participant. Billy was the only student to give a Stage I response on the seriation posttest. The other students gave Stage III responses on the seriation posttest. The seriation task was administered to determine the perceptual discrimination and organizational skills of the participants. Perhaps Billy's difficulty with programming was due to the lack of these prerequisite skills.

Billy's answers to the student interview instrument revealed his frustration with programming. When he was asked what he liked about the computer, he replied that he liked playing the preskill game "G," the object of which was to get the Turtle in the box. Most students liked typing and making pictures on the computer. When he was asked if he liked making pictures on the computer, he said, "I think that's hard to do. But I don't feel like it will be very much fun at all."

On the parent questionnaire, Billy's mother did not circle "competence" as a feeling which her son had expressed at home in regard to his work with the computer. She noted that he had remarked that "making the Y was hard." She did
circle "happy" because Billy enjoyed doing the preskill games on the computer.

What can be concluded about the programming competence of the participants from their responses on the instruments? The results of the student interview and the parent questionnaire indicated that Billy did not enjoy programming the computer and the programming tests revealed that he was only able to write one of the three shape procedures and missed two of the four dot tasks. Billy was not a competent programmer. He liked to play the preskill games on the computer. They were simple and did not require any organizational skills.

Danielle and Jonathan did very well on the programming tests. They were independent programmers. They were successful in writing workable procedures for the three shapes.

Alexandra and Andrea were also programmers, but they were not as independent. They relied on turning the Turtle ninety degrees and did not attempt to try other inputs when ninety did not work in their triangle procedure. They may have had a perceptual discrimination problem in noticing angle differences.

According to the responses on the parent questionnaires, all of the participants had a positive experience with the computer. Some of the students were more independent programmers than others. Only one student could not be considered a programmer.
The results of two instruments, the screening test and the conservation of number Piagetian tasks, indicate that student understanding of number quantity and equality improved during the study. Billy and Jonathan corrected their pretest errors on the relative number quantity section of the screening test when it was given again at the end of the study. Alexandra and Jonathan shifted to Stage III, conservation of number, on the posttest of that task. They were no longer confused by the various physical arrangements of the chips. Instead they counted each set to prove they had the same number of chips. In both tests, students were required to compare two quantities and determine whether one had more or whether they were equal. Three students demonstrated better understanding of relative number quantity and equality by the end of the study according to the results of the instruments.

Field Notes and Transcriptions

The major body of data collected during the study was comprised of the transcriptions of the tape recorded daily sessions with the students and the researcher’s field notes of the commands students entered into the computer and other observations about their behavior. Several copies of this data were made. The data were categorized according to five types of student behavior exhibited while they were working with the computer. Each of the five areas is analyzed in
the following section. The difficulties which students experienced as they programmed the computer are discussed below. In the interest of presenting a balanced perspective of the programming abilities of young children, the strengths they exhibited in each area are discussed as well as their difficulties.

**Conceptual Difficulties**

Some of the participants had difficulty understanding certain mathematical concepts which related to programming in LOGO.

**Place value.**—Several students had difficulty understanding place value. Alexandra said she wanted to type in the number "fifty-five." She typed 505, a logical mistake when the first two digits are read together. Sometimes students knew the number input they wanted, but because they did not understand place value, they did not communicate what they wanted clearly. Danielle said she wanted the number "one ten two" and wrote 112. During another session when a number greater than ninety was needed, she suggested "ten hundred." She was referring to the number 110. At other times students typed in the number they desired, but they could not name it correctly. Jonathan typed FD 678. When he was asked what number that was, he said, "sixty-seven hundred and eight."
Terms.--Students encountered some terms which needed to be explained to them during their programming experience. The following words were unfamiliar to some of the participants: numeral, side, and rectangle.

Andrea asked about the meaning of the word numeral while she was looking for the letter T on the keyboard.

AN Can't find T.
R It's not on the row with the numerals.
AN What's a numeral?
R Numerals are all these numbers here on the top row. There aren't any letters up here, just numerals.

When Billy was asked how many sides a square had, he could not answer because he did not understand the concept of side. However, when a square was drawn on paper, he could point to the sides and count them.

Andrea had a very limited conception of the meaning of rectangle. When she encountered a figure with long horizontal sides which the researcher called a rectangle, she said, "How do you know? It doesn't look like a rectangle. Rectangles look like doors." Her concept of a rectangle needed clarification.

Coordinate geometry.--The Cartesian coordinate system of coordinate numbers on a grid was explained to Danielle when she was trying to move the Turtle to the left corner of the screen in order to begin her name program. The LOGO command SET POSITION and two coordinate numbers placed in brackets moves the Turtle to the designated position on the
screen. Danielle did not understand the coordinate system, the coordinate numbers representing a point in space, or negative numbers. These concepts were too abstract for her to comprehend.

**Conceptual Strengths**

**Number quantity.**—Students developed an excellent understanding of number quantity. At the beginning of the study they selected number inputs which moved the Turtle forward only a short distance. Their estimates included 1, 7, 9, and 13. As the study progressed, students learned that larger number inputs such as 40, 60, 90, and 100 caused the Turtle to go farther across the screen. As she typed in a long line of numbers, Andrea remarked, "It would go real farther if I did that!"

The researcher employed the practice of labeling numbers as big or little and taught the participants to do so as well. (A refers to Alexandra and R refers to the researcher.)

R You know why the Turtle didn't go very much? What kinds of numbers are these? (pointing to LT 3, LT 5, LT 9)
A small numbers
R And what kind of a number is this? (pointing to RT 130)
A big number

Sometimes students created their own semantic labels for numbers such as "a high number," "a long number," or "a little, tiny number," but the function of distinguishing between number magnitudes remained the same.
Students learned to vary the size of the number input according to their programming needs. They became adept at determining the relative size of numbers and chose appropriate inputs as they worked out the errors in their programs. The following examples illustrate their understanding of number quantity.

R  You tried ninety and you thought that was too big.
D  So I'm gonna try seventy! Yeah.
A  One and a sixty?
R  Let's try that one. (Alexandra types 160.) What do you think?
A  I think that's too high a number.
R  Yes. Think of a number between ninety and 160.
A  130. (types 130)
R  Okay. Ninety didn't turn enough; 130's turning too much. So we need a number that's a little bit smaller than 130.
A  One and two zeroes.
R  Do you know what that number's called?
A  One hundred?

Students saw differences in the relative quantity of numbers demonstrated on the screen by the Turtle in the distance it traveled and the amount it turned, dependent on the number inputs. They became skillful at selecting numbers of the proper magnitude.

Summary.—Participants had difficulty understanding abstract concepts such as place value and coordinate geometry. Terms which could not be represented in a concrete way were also difficult for students to comprehend. However, students did develop an understanding of relative
number quantity during the study. They labeled numerals as big and little to help distinguish their relative magnitude.

Perceptual Difficulties

Students experienced a number of perceptual discrimination difficulties due to their developmental immaturity. They had problems with visual discrimination, orienting in space, and auditory discrimination.

Visual discrimination. Three types of visual discrimination difficulties occurred during the study: perceptual "overload," discrimination among letters and numerals, and reversals. One student mentioned that she had trouble finding the keys she wanted when she was typing in a long command, REPEAT 360 [PD .5 RT 1]. She thought that writing it on paper was easier.

AN I write [sic] this at home. Every bit.
R You wrote it down at home? I could tell you remembered it.
AN It's kind of hard to find them because there's so many keys. But when you're at home and you just write it down, it's not so hard.

Andrea experienced a visual perception "overload" as she searched for one key at a time among many on the keyboard.

All of the participants had difficulty discriminating among letters and numerals which had similar shapes. The ones which were most frequently confused are listed in Table V. All students had to be taught the computer's symbol for zero (Ø). The visual discrimination problems occurred
throughout the study, not just at the beginning. Sometimes Andrea confused the use of the CTRL (Control) key for the SHIFT key when she made brackets. (Brackets are created in Apple LOGO by typing SHIFT N and SHIFT M.) Both the SHIFT and the CTRL keys are larger keys with several letters on them and are used for special purposes. She probably confused the functions of the keys as well as their appearance.

TABLE V

VISUAL DISCRIMINATION PROBLEMS WITH LETTERS AND NUMERALS

<table>
<thead>
<tr>
<th>Typed</th>
<th>Intended</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>M</td>
</tr>
<tr>
<td>M</td>
<td>W</td>
</tr>
<tr>
<td>0</td>
<td>Ï</td>
</tr>
<tr>
<td>I</td>
<td>1</td>
</tr>
<tr>
<td>Q</td>
<td>0</td>
</tr>
</tbody>
</table>

Another visual discrimination problem was typing in letters or numerals in a reverse order. Students never noticed their own reversals; they had to be called to their attention for correction. Reversals occurred because students would read the command or think of it from right to left and type it that way. All of the participants had reversal problems throughout the study. Table VI contains a list of common reversals typed during the study and what students intended to type.
Students may have made reversals simply because they did not realize that proper letter order was important. The following quotation from a conversation with Andrea during the eighth week of the study indicates that she thought entering the correct letters, regardless of order, was all that was necessary.

R You've got F-D and then you have D-F.
AN They're the same thing.
R This one and this one?
AN I mean they're the same letters.
R They are the same letters, but they have to be in the same order.

Occasionally students reversed whole units or an entire word. They typed 21 FD instead of FD 21 and LONG SAVE instead of SAVE LONG. One student typed in the name of his program backwards because he was reading it from right to left.

Although the problem of reversing letters and numerals is
normal for five-year-olds, it was frustrating for the participants because they had to delete the errors and retype carefully.

**Spatial orientation.**--Students frequently confused the directions of right and left. They relied on the labels denoting RT and LT which they wore on the backs of their hands. Danielle remarked, "I need the stickers cause I keep forgetting."

Another common problem was adopting the perspective of the Turtle in space as they decided which direction to turn it. If the Turtle's "head" was pointing to the top of the screen, students just held up their hands near the screen and matched the appropriate command with the way they wanted the Turtle to turn. However, if the Turtle was pointing down, students had to reject their own egocentric perspective and try to assume the spatial orientation of the Turtle as they decided which way to turn. This was difficult for all students. They were only able to understand it by standing up, turning around to face "down" like the Turtle, and checking the labels on their hands for the name of the direction they desired. The excerpt below illustrates the problem.

A  (types LT 90)
R  Let's just see what he does. Is that the way you want it?
A  (She shakes her head no.)
It looks like it would go left, doesn't it? But that only works when the Turtle's head is pointing up. When the Turtle's head is pointing down, you have to turn the way the Turtle is. See? He's facing this way. (researcher turns) He turns to the right. Stand up and do it with me.

After several experiences with this type of situation, students were able to figure out the problem independently. When they encountered the Turtle "pointing down," they would stand up and realign themselves with the Turtle or they would merely turn around in the chair and face the other direction. Three students did this without prompting during the programming tests and selected the appropriate turning command.

Some students found it difficult to believe that the Turtle moved when they gave it small inputs because the movement was so slight. This was a very minor problem during the study. Jonathan seemed skeptical that the Turtle moved when he typed RT 4. (The movement was almost imperceptible.) When he typed FD 3, he said, "It don't look like it moved." Students focused their attention on finding the correct keys as they were typing and often did not shift their eyes to the monitor in time to see the Turtle moving in response to their commands.

Auditory discrimination.—Four of the participants confused numerals which had names containing similar sounds throughout the study. Some examples of numerals which were
confused are provided in Table VII. The numerals the students said are on the left and the corresponding ones they typed or desired are on the right of the table.

**TABLE VII**

AUDITORY DISCRIMINATION PROBLEMS WITH NAMING NUMERALS

<table>
<thead>
<tr>
<th>Pronounced</th>
<th>Intended</th>
</tr>
</thead>
<tbody>
<tr>
<td>13</td>
<td>30</td>
</tr>
<tr>
<td>14</td>
<td>40</td>
</tr>
<tr>
<td>15</td>
<td>50</td>
</tr>
<tr>
<td>16</td>
<td>60</td>
</tr>
<tr>
<td>70</td>
<td>17</td>
</tr>
<tr>
<td>80</td>
<td>18</td>
</tr>
<tr>
<td>90</td>
<td>19</td>
</tr>
<tr>
<td>114</td>
<td>140</td>
</tr>
<tr>
<td>140</td>
<td>114</td>
</tr>
</tbody>
</table>

Students mixed the sounds of the numerals thirteen through nineteen and the tens thirty through ninety. In addition to confusing words of similar sounds, Jonathan had difficulty distinguishing between some letter sounds. The letter sounds of **T** and **D** were particularly difficult for him. In the following example Jonathan was typing in the unfamiliar word, SETHEADING.

J (spelling as he types) H-E-A
R D. You passed over the D. No, you want D first.
   D-I-N-G.
J T-I-N-G.
R You don't need a T, Jonathan.
J D-I-N-G.

The fact that he was unable to read or understand the word precluded any chance of his noticing the error in context.

Another problem students encountered was discriminating between two words which sounded the same but which had different meanings. Two such homonyms were TO (used before the name of a procedure) and the numeral two.

R So what I want you to do is type TO BOX:
   B-O-X.
J Two numbers?
R The word to: T-O.
J T-O.

Students were more accustomed to the sound of the word to referring to the numeral two. Therefore they often typed the numeral two first when they were asked to type TO.

One of the LOGO commands also created some confusion because it was a word in common usage as well as a specialized command. That word was right. Andrea differentiated between the two meanings as she played with the words.

AN RT?
R Yes.
AN RT sounds like I'm right but I have to push RT. It sounds so funny.
R And I say "right."
AN And that's right, I have to push RT. Space, right?

Sometimes students had difficulty distinguishing between the use of right to confirm a correct action or statement and RT used as a turning command because they had the same sound.
Summary.—Students' visual discrimination problems included (1) discriminating among the many choices of keys on the keyboard, (2) discriminating the salient features of certain letters and numerals which were similar in appearance, and (3) discriminating the correct order of letters and numerals. In the area of spatial orientation participants had difficulty remembering left from right, adopting the perspective of the Turtle, and seeing any movement when small inputs were given. Students' auditory discrimination problems involved distinguishing between (1) numeral names with similar sounds, (2) letter names with similar sounds, (3) the correct usage of **TO** and **two**, and (4) the meanings of **RT** and **right**. All of the students experienced perceptual difficulties as they worked with the computer during the study.

**Physical Difficulties**

Students experienced some difficulties in programming because of their physical impulsiveness and unusual finger- ing approaches to typing. Some typing errors were due to a lack of coordination.

**Posture.**—Andrea favored a slumped down position in the chair in front of the computer. Sometimes she had to be reminded to sit up so she could see the letters and type accurately. One day while she was slumped down with her legs stretched out under the table, she began kicking and jarred the wires connecting the computer to the electrical
outlet. LOGO had to be rebooted on the computer as a result of her actions. Danielle also kicked her feet under the table against the wires on one occasion.

**Fingering**—Students used a variety of approaches to typing on the keyboard of the computer. Jonathan was the only participant who used his thumb in typing. All students relied on their index and third fingers to do most of the typing. Alexandra and Danielle used those fingers on both hands (usually one at a time) to type commands. Alexandra used her fourth finger at times to press the keys, and Danielle used her fifth when she was playing the preskill games. Danielle remarked, "I use my pinkie for L when I only need to push one."

Students typed with their second and third fingers with considerable facility and accuracy. The speed of their typing improved during the study.

Andrea was the most creative in inventing new approaches to pushing the keys. She rolled her knuckles over them. She bounced her thumb off the space bar. She typed with her second and third fingers simultaneously (two fingers per key). Her most unusual approach was bending her fingers backward with the left hand and flipping them to hit the keys. She made many typing errors because her methods were not very accurate.
In contrast, Alexandra and Billy were very conservative in their approach to typing. They both typed slowly and tried to avoid making errors. They pushed each key precisely. They were usually deliberate, careful typists.

Several students adopted the approach of spelling the word after the researcher had written it and respelling each letter in turn as they located and typed them. Alexandra employed a systematic strategy in locating the letter she wanted on the keyboard. She said the letter, then slid her index finger from left to right across each key in every row on the keyboard. Her eyes followed her finger until the correct key was found and pushed.

**Typing errors.** Some typing errors were caused by a lack of physical coordination. The CTRL key was used in conjunction with other keys to make the computer perform special functions. For example, CTRL and G makes a program stop immediately. Students pushed those keys when they wanted to stop one preskill game and play another. At the beginning of the study, students experienced difficulty in keeping the CTRL key pushed down while they pushed G with the other hand. They pushed the two keys successively instead.

Sometimes students made typing errors because they held keys down when they should have pushed one at a time. Andrea
played the computer's keyboard like an organ at first, with each finger holding down a different key.

Several students made typing errors because they tried to push two keys simultaneously with different fingers, which sometimes resulted in reversals on the monitor. Occasionally two-digit numerals were typed in this way.

When excessive force was used in typing, sometimes the finger bounced on the key and resulted in a repeated letter or numeral. Andrea and Danielle did not understand why a duplicate appeared on the screen, because they thought they had only pushed the key once (hard). Students were able to moderate the force they applied to the keys when they wanted to.

Jonathan made some typing errors because his finger hit the space in between several keys and not on top of a single key. Consequently, several letters appeared on the monitor instead of one. This action was probably a combination of poor coordination and carelessness.

At times students forgot to push the space bar between the command and the number input. An error message resulted and the line had to be retyped. Students became adept at identifying their errors. When Jonathan typed FD85 incorrectly, he said, "Oh, oh. Forgot to put a space there."

Students often got in a hurry and typed the command without the number input or typed the number without the
command. They did not check the monitor before they pushed Return, and therefore received an error message.

**Summary.**—Although some typing errors could be attributed to perceptual difficulties, many typing errors were caused by the impulsive actions of the participants. Some errors were caused by the playfulness of the participants experimenting with new techniques for pushing the keys. These playful and impulsive actions were natural kinds of behavior for five-year-old children. Poor coordination was also a cause of typing errors. Individual differences existed in physical maturation and impulse control. Some students were careful, systematic typists most of the time. Other students exerted less self-control and were more physically active as they typed on the computer.

**Affective Responses**

Students displayed many types of emotions as they worked with the computer. Inferences about their feelings were made from what the participants said and did during their daily individual sessions with the computer. Sometimes students conveyed their feelings by talking to the computer or to the Turtle.

During the study the participants expressed preferences for certain keys and numerals. They had favorite commands, typing actions, and preskill games. Students exhibited
individual creativity in their programs and the program names they selected.

**Positive feelings.**—Several students described their work on the computer as being "really neat." Billy said, "It's fun working on the computer." Students had positive feelings when they completed a program or were successful in getting the Turtle in the box in the preskill games G or GAME. After students had accomplished a goal, they made comments such as, "This is easy to do," "Got that right," and "I want to do it again." When Jonathan felt proud of his work on the computer, he said, "I wanna write 'Good Work.'" He wanted to reward himself with a message of praise from the computer.

Instances of student laughter were interpreted as expressions of positive feelings. Students laughed when their work on the computer was successful, when it produced unusual results, and when their predictions were confirmed.

Andrea was delighted at the result when she typed in the number input of 100 in the preskill program, GAME. She laughed, kicked her feet, and made sounds like baby talk.

Students also laughed when their typing produced unexpected results. Billy laughed when he moved the Turtle to the edge of the screen. Half of the Turtle was visible and half was off the screen. He said, "It looks like somebody took a bite out of the Turtle."
Students enjoyed watching the Turtle go off the edge of the screen and reappear. This occurred when Andrea typed in commands which made two circles larger than the screen. The Turtle commenced the curve of the circle, went off the screen and returned at a different place on the screen seconds later. Andrea laughed and said, "That's funny. Went off the screen! [Turtle reappeared.] There we go." Students intentionally typed in large number inputs such as 4560 to cause the Turtle to wrap the screen. The Turtle made straight lines across the screen to the edge and reappeared at a parallel point on the opposite side of the screen. Large number inputs would result in parallel lines spanning the screen. Most of the students enjoyed making the Turtle wrap the screen and laughed at the wrapping effect.

Students also liked using the REPT (Repeat) key. Jonathan laughed when he saw a row of Fs result from his pressing the F key and the REPT key at the same time. The participants were surprised at the effects created when the command REPEAT was typed in conjunction with a numeral and the name of a program. The Turtle drew the graphics commanded by the program the designated number of times. Interesting patterns were drawn on the monitor in this way. Students were delighted in the graphics made by repeating their programs and often laughed at the results.
Several students typed commands incorrectly on purpose so that error messages would appear. They seemed to be testing their predictions and the consistency of the computer. The following example illustrates this type of behavior.

D Hey, watch this.
R What does it say?
D (laughs and reads the error message)
"I don't know how to H-D-T."
R That's right. And you knew it would do that, didn't you? (D nods.) You're really trying to mess up this computer, aren't you?
D Wait, I want to do it. (types another error)
R To see what happens?
D Yeah.
R All right.
D (laughs)
R You did that yesterday, too, didn't you? Made mistakes on purpose!
D Yeah.

Students enjoyed making mistakes on purpose because they felt successful when the result they expected appeared on the monitor (the "I don't know how to . . ." statement) and they liked reading the messages. Their laughter expressed their positive feelings when they saw their predictions confirmed.

In general, the participants enjoyed their work with the computer. They were eager to leave their kindergarten classrooms every day to work with it. Positive feelings were generated by the success students experienced when they played the preskill games or programmed the computer. Their laughter indicated surprise and pleasure at seeing unusual
results on the monitor or satisfaction in having their predictions confirmed.

Negative feelings.--There was a noticeable difference in the frequency in which negative feelings were verbalized by the participants. Alexandra and Jonathan did not express any negative feelings during the daily programming sessions. Andrea and Danielle each made three negative remarks. Billy made comments which expressed reluctance, lack of self-confidence, or frustration nineteen times during the study. These individual differences in student attitudes toward programming may have been related to their success in programming, their task perseverance, and their self-confidence.

Billy was the only student who was reluctant to touch the computer on the first day of the study. He refused to turn on the monitor or the computer after he watched a demonstration of these actions. However, he did push the keys F, B, R, and L in the preskill game DRAW later during the session.

Billy was also reluctant to try new programming tasks. When he was asked if he would like to make a square like the one the researcher had made on the monitor, he replied, "I think it will be too hard! It's not hard for a big person [adult], but it will be too hard for me." Billy was more receptive to the idea after the researcher suggested that first he draw the square on the playground with chalk and
then count the number of steps on each side. He said, "Well, maybe I can. I'm almost six, you know. Let's do that right now."

Billy lacked self-confidence. He did not expect to succeed in programming and it was difficult to convince him to attempt the programming tests. The following quotations illustrate Billy's negative attitude and the researcher's attempts to keep him on task.

**Dot Task 1**

B I think I can't do this.
R I think you can. (Billy types several commands.) How many more steps, do you think?
B I don't know. It can't get on the dot. I just can't understand how to do this stuff. It's just hard!
R You're doing very well, Billy. Look where that Turtle is. I think you're very close to that dot. (one step away)

**Dot Task 2**

R We still have to finish the box here.
B All right, so we have to push . . . I don't think I ought to do the rest of this. I'll do dots--the four dots.
R But we can't go on to that posttest until we put the other side on this box.
B All right, now push . . . there! Got that right.

**Dot Task 4**

(before beginning dot task)
B This will take forever to do it.
R You think it will?
B Oh, I don't know how we can make it up there.
(Referring to the forty-five degree angle)
R Just do the best you can.
B Oh, God! Another boy knew how to go up here?
R You're the first one to try this.

Billy became easily frustrated when he worked on the computer. Whenever the graphics he made did not please him, he gave up or depended on the researcher to tell him what to
type to fix the problem. He did not act as though he were in control of the computer.

R Is that the way you want it to go?
B Yes. Whichever way it wants to go. Now forward of it. [He pushes F several times in DRAW.] All gone now. [The Turtle goes off the upper edge of the screen.] Well, tell me what I'm gonna push!
R What did you push to make it go back?
B It went way up there and now it's gone.
R (backs it up)

If he encountered a problem, Billy rarely worked on the task until the problem was resolved. When he moved the Turtle to the edge of the screen, he didn't know what to do. The Turtle's image was split and each half was on opposite sides of the screen.

B I guess that's just how it is.
R What would you like it to do, Billy?
B Where do you want it to go?
R I'm finished drawing now.
R Would you like to try a different picture?
B You can. (R clears the screen.)
B This is the last one that I'm gonna do.

Billy solved computing problems by clearing the screen and starting over or by leaving the situation. When Billy felt unsuccessful in his computer work, he often asked to return to his kindergarten class. He was always allowed to do so. Leaving the source of frustration was one of the ways he avoided feelings of inadequacy.

Andrea and Danielle were also frustrated at times in their work with the computer. They did not understand why certain commands had to be retyped.
Every program stored on a minidisk had to be loaded into the computer's immediate memory before it was operational. Then the name of the program had to be typed so the computer would execute the commands.

When Andrea was working on the size of the tires in her SMOKY program, she had to type REPEAT 360 and insert various number inputs in the brackets which followed the command in order to find a circle the size she desired.

Danielle expressed her frustration with physical force on one occasion.

The types of negative feelings the students experienced and the frequency of their occurrence varied during the study. Billy was reluctant to take risks and try new programming challenges. He exhibited a lack of self-confidence and
a low tolerance for frustration in his programming attempts. Billy made comments which indicated that he did not feel in complete control of the actions of the Turtle. He did not try alternatives when he encountered programming difficulties. Rather he chose to rely on directions from others, to start over, or to leave the situation.

Andrea and Danielle expressed frustration when they had to retype certain commands. This action seemed redundant to them. They did not comprehend why some commands had to be typed again. On rare occasions frustration was vented on the frame of the computer in blows accompanied by verbal insults.

Talking to the computer.---Students directed remarks to the Turtle or to the computer (or parts thereof such as the monitor or the disk drive). They talked to the computer when they were frustrated, when they exulted in their accomplishments, and just before they left the room. All of the students except Jonathan talked to the computer during the study.

Danielle talked to the computer several times when she was frustrated. When her program for the letter A was not drawn where she wanted it on the screen, she scolded: "Now you Turtle! Why don't you straighten up?" She was impatient while waiting for the disk drive to load a program and said, "Come on, you disk!" Mechanical problems were a source of
frustration. When the monitor malfunctioned and caused wavy lines, Danielle said, "That looks so funny. Stop it, computer! Stop it, stop it! Oh, oh. What is that thing doing there?" Andrea ordered the Turtle to stop when it went off the screen as it made the tire for her truck program.

Sometimes students talked to the computer in order to gloat about their programming success. After Billy turned the Turtle correctly in one of the posttests, he exclaimed, "Got you this time, computer!"

Danielle felt a sense of mastery over the computer. She enjoyed calling it stupid because she realized that her intelligence was superior and the computer merely followed her commands. When she typed in an error on purpose and received an error message, she giggled, "You stupid thing!"

One time she tried to address a command directly to the Turtle by typing in T for Turtle.

D Watch this: Oh, Turtle, go forward.
T for Turtle.
R (laughing) No! He knows you're talking to him. You don't tell him T for Turtle! He won't know what it means.
D Hey, Turtle! Are you there? I'm talking to ya! (laughs) GO, space, BK 76. [She types in the commands as she says them.] Now, watch this. Hey, you stupid thing!
R It did go back, didn't it?
D I know.

Students also talked to the Turtle just before they turned off the monitor and left the room. When the researcher asked Billy to turn off the monitor, he responded, "Good night, Turtle" before he pushed the knob. Alexandra
had worked for several days on a program that drew a picture of a mouse. At the end of a session, she was asked to clear the screen. As she did so, she said, "Good-by, Mouse. Bon voyage!"

Students' remarks to the computer may be categorized according to the following purposes: they implored the Turtle to do what they wanted; they protested mechanical problems; they insulted the computer; and they told it good-by before they left.

Attention span.--The amount of time in which students devoted their full attention to programming varied depending upon their physical and emotional state. Class activities which especially interested the participants also reduced their attention span.

Sometimes students were anxious to leave so they could resume an activity in their kindergarten classroom. Billy ended one session by saying, "Let's go so I can work with the clay." When Jonathan was asked if he would like to make any changes in his program, he replied, "No, I want to finish this tomorrow so I can finish my book." He had been making a book about Halloween in class before the programming session began.

Snacks were a powerful distractor. Snacks were served to the kindergarten classes outside during recess. Students were anxious to rejoin their class before the snacks were
distributed. The following incident indicates their concern that they be back in time for snacks.

J Hey, you know what Alex brung? Well, he brung some cupcakes for snacks because it's his birthday today.
R Okay. I'll get you back in time for snacks.
J Why can't you turn on that clock? (refers to timer)
R Well, we can. Let's work for about five more minutes and then we'll get back for snacks, okay?

Jonathan wanted to insure that the researcher would close the session in time, so he asked that the timer be set.

Students' physical needs had to be met in order for them to devote their full attention to programming. Andrea was restless and was not able to concentrate when she worked on the computer just before recess. During the last hour of kindergarten she was too fatigued to pay attention in the programming sessions. Therefore she worked with the researcher during the first half of the morning. Sometimes students seemed distracted because they had to use the restroom. In one case a student was inattentive because she had taken medication for allergies. She was too drowsy to finish her session with the computer.

The emotional needs of students were also a factor in the length of time they worked with the computer. Very little was accomplished during one session with Danielle because she was so excited. It was the last day the student teacher would be with the class and Danielle wanted to make
her a card. After a few futile attempts at programming, she made the card instead.

The programming tests were mentally fatiguing and frustrating for some students. Billy completed the first dot task in eleven minutes. The following incident ensued when the researcher encouraged him to complete the second dot task as well.

R  Do you want to try another dot-to-dot?
B  No, I think it will take too long.
R  Here's an easy one. Watch.
B  I don't think so, I don't think so.
R  Let me show it to you first and then you can decide, okay?
B  (sees it) How can I do that? I don't think I can do this. I don't feel like to do it.
R  Okay. We can try it another time.
B  Woof! (barks like a dog)
R  Why did you do that?
B  Cause I want to get out of here! I want to go back to my class.

Billy "barked" several times during the study to signal his frustration with programming and his desire to leave the situation quickly.

Danielle had worked for fifteen minutes on the third shape procedure for a triangle when she remarked, "I'm tired of working with this computer. Did you know that?" She was frustrated because she had just discovered that RT 90 would not turn the Turtle to the angle she wanted in her triangle. She was willing to work for five more minutes on the procedure because she wanted to complete the task ("I think I want to do the rest of that.") Students were more likely to
give up their programming attempts when they were dis-
couraged and mentally fatigued.

One of the research questions was "How long can the computer hold the attention of five-year-olds?" In an effort to answer that question in a quantitative way, the following procedure was utilized. One copy of the transcriptions was divided into five notebooks. Each notebook contained all of the dialogue spoken and programming done by one child during the daily sessions. The longest transcribed session in which students were involved in programming (not playing preskill games) was isolated in each notebook. The tape recording of that particular session was played and timed with a stop watch. Table VIII provides the length (in minutes) of the longest programming session for each student.

<table>
<thead>
<tr>
<th>Student</th>
<th>Longest Session</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alexandra</td>
<td>25</td>
</tr>
<tr>
<td>Andrea</td>
<td>20</td>
</tr>
<tr>
<td>Billy</td>
<td>14</td>
</tr>
<tr>
<td>Danielle</td>
<td>35</td>
</tr>
<tr>
<td>Jonathan</td>
<td>20</td>
</tr>
</tbody>
</table>
Students were encouraged to work on the computer a minimum of twenty minutes per day. However, if they were unable to sustain their attention or interest in programming for that length of time, they could play preskill games on the computer or return to their classes. If students were engaged in programming at the end of the twenty-minute time period, they were allowed to continue until they reached a "stopping point" or until they became distracted and displayed off-task behavior. When student attention waned, they were informed that another student was waiting to work with the computer and their time was up.

According to Table VIII, student attention spans varied greatly. Two students, Alexandra and Danielle, were involved in programming approximately twice as long as Billy. Andrea and Jonathan sustained their attention for twenty minutes while programming. Danielle's time on task was ten to fifteen minutes longer than the other students.

Although some difference existed in the length of attention given to programming during the longest session, student attention varied on a daily basis and was dependent on several factors. Student interest in class activities such as snacks sometimes interfered with their ability to concentrate fully on programming. Physical needs for movement, elimination, or rest interrupted or cut short programming sessions. Intense emotions such as excitement or frustration affected the amount of time students devoted to
programming. At times students experienced mental fatigue and wanted to stop working. These factors or a combination of them affected the length of time in which students concentrated on programming.

**Expressions of independence.**—Four of the students indicated that they did not want the assistance of the researcher during the study. Jonathan was the only participant who did not express the desire to work independently.

Students rejected attempts by the researcher to spell commands such as END for them. Andrea said, "Don't tell me. I'm gonna close my ears." Students wanted to demonstrate their ability to spell commands without assistance.

R  Now I want you to type in LOAD "GAME. Do you remember how to spell LOAD?
B  Yes. (types LOAD)
R  You do! Space.
B  I know! I know how to do that!
R  I won't say any more then!

Students protested vigorously when the researcher offered to help with a task that they were capable of doing independently. Sometimes they needed more time to recall and spell the commands than was expected.

R  Let's hide the Turtle and see how your picture looks. (waits for a response) H-T.
D  I know. I was just remembering.
R  Excuse me.

Silence did not always indicate lack of knowledge; rather, it was essential thinking time.
Once students had worked out the commands for their graphics, the commands had to be typed in again under the name of a program so that they could be saved onto the floppy disk. At first, students wanted the researcher to dictate the list of commands so that they would not have to locate the next command on the paper and then type it in. However some students decided that they no longer wanted the researcher to dictate the commands. They preferred to type in their programs without assistance. Andrea insisted, "I can remember all of these things [commands]."

The participants seemed to take pride in working alone. Alexandra worked out the commands for her graphics silently near the end of the study and seldom consulted the researcher. Students liked to demonstrate their independence by refusing to rely on the visual aids provided by the researcher. Andrea felt that she no longer needed the hand labels designating RT and LT in the seventh week of the study.

R We need some stickers, don't we?
AN How do you know I need them?
R Cause you're going to make the Turtle turn in a minute, and I think these stickers will help you know which way to make it turn.
AN No, put them back. I know.
R The Turtle's head is facing this way.
AN I know. Put them back.
R Okay. Now tell me which way the Turtle should turn.
AN Right.
R You did know.
AN Told you.

Alexandra was delighted when she succeeded in making brackets on the computer without consulting the chart which
illustrated how to do so. She brought the fact that she did it by herself to the attention of the researcher: "I didn't look at that [chart]." She was proud of her accomplishment.

Sometimes students protested when the researcher intervened to type new commands quickly. When Danielle wanted to see the commands for one of her programs, the researcher typed PO (Print Out) and the name of her program. This was done to save time because Danielle had not yet learned the command PO and the immediate goal was to review the commands in her program. Danielle protested politely by reminding the researcher that she was also a fast typist.

R See the program again. Print Out DRAW5.
(typed PO "DRAW5")

D I bet I could put DRAW5 better than you could--faster, I mean.

R To type it in? Okay. I'll let you do it next time. I'm sorry.

Students insisted that they be allowed to do the things which they were capable of doing without assistance. This striving for independence is characteristic of young children. They knew when they no longer required aid with spelling, typing, and turning directions and communicated that freely to the researcher.

Individual preferences.--Students were given a great deal of freedom in programming and typing. As a result, they engaged in some behavior which they enjoyed but which the researcher did not expect. Students demonstrated
individual preferences for particular keys and commands. They used some numbers more frequently than others in their programming. Students enjoyed repeating certain actions. They were allowed to select the preskill games they wanted to play, and they had particular favorites.

The three girls in the study were fascinated by the punctuation marks on the keyboard of the computer. Danielle typed in lines of periods and commas. Andrea asked, "Will it understand that?" after she typed a slash, a colon, a semicolon, a period, and a comma. Alexandra typed all the non-letter combinations which could be made with the SHIFT key and the numeral keys. When she was asked what she was typing, she replied, "The neat things." She also liked to make brackets with SHIFT N and SHIFT M.

Students enjoyed pushing the REPT (Repeat) key and another key at the same time to create several duplicate lines of the character.

AN What did I do?
R You pushed REPT and 2, and it made a bunch of them.
AN I want to see how it does that. PETE.
R REPEAT--That makes it do it over and over again.
All those twos.

Some students preferred certain letters because they linked them to something which was meaningful to them. Danielle said, "I like the letter L because it's in my name."
Andrea liked the RETURN key because it made the top command disappear when the lower part of the split screen was full.

AN Hey! It took one of my PDs and fives away.
R It did. It scrolled off the edge of the screen there.
AN Oh, because of that (pushes RETURN). I push it a couple more times. It's gonna go off the screen!

She liked the disappearing effect of the RETURN key.

The forward arrow key was a favorite of all of the participants. This key is operational in the EDIT mode; it moves the cursor one space to the right. However, in the immediate mode, it has no function except to make a beeping sound. The students discovered the novel sound and liked to push it repeatedly. Andrea said, "I like it when it goes like that: beep." Sometimes they pushed the forward arrow key and the REPT key to make it beep for several seconds.

Certain commands were very popular and were used frequently by the participants. Students reacted positively to the commands HT (HIDETURTLE) and ST (SHOWTURTLE). When Jonathan typed ST he said, "I want to make him [Turtle] come back." They liked seeing the Turtle disappear and reappear.

The command CS (CLEARSCREEN) erased the graphics from the screen. Students enjoyed making their pictures disappear by typing CS. Danielle became adept at using SETH 0 (SETH-HEADING 0) in her programs to make the Turtle point to the
top of the screen. The use of the command BK (BACK) varied among the students. Both Alexandra and Danielle employed BK frequently in their programs. They each typed BK with twelve different number inputs. In contrast, neither Billy nor Andrea used BK in any of their programs. Three students preferred to turn the Turtle with a LT (LEFT) command rather than a RT (RIGHT) command. Alexandra typed RT turns more often than LT turns.

Students also used certain numbers frequently in their programming. A frequency count was taken of all the numbers used by each student in his or her computer programs and in the programming tests. Numbers which were used by the student a minimum of five times are included in Table IX.

TABLE IX

NUMBERS MOST FREQUENTLY USED IN PROGRAMMING

Alexandra . . . . . . . . 3, 4, 5, 10, 20, 40, 50
Andrea . . . . . . . . . . 1, 5, 12, 21
Billy . . . . . . . . . . . . 10, 27, 30
Danielle . . . . . . . . . 9, 10, 20, 40, 70
Jonathan . . . . . . . . . 3, 100

Alexandra preferred to use tens in her programming. Andrea did the opposite. The numbers she used in her commands rarely ended in zero (except for RT and LT 90). She transposed the order in some of her inputs such as FD 12 and FD 21. Billy's cautious approach to programming was exemplified in his
choice of numbers. He never selected a number greater than forty-three (except when he used RT or LT 90 for turning). Jonathan used three as a number input fourteen times in his programs. Danielle typed the number forty fifteen times in her programs. During one programming session, Andrea typed FD 5 thirteen times in succession.

R  Can you tell me why you always push FD 5?
AN  Because that's the easier number.
R  Why is it easier do you think?
AN  Cause it's just easier for me.
R  Okay.
AN  I like it.

Students had definite preferences for certain numbers. Billy used the number ten almost exclusively during the first three dot tasks. In the third task, he created the rectangle by giving a series of six FD 10s for the longer sides and three FD 10s for the shorter sides. When he was asked why he typed in FD 10 so much, he answered, "I like to." The inputs of FD 10 and FD 5 moved the Turtle short distances and probably gave the participants a feeling of control.

Students repeated particular actions during the study which they enjoyed and were meaningful to them. Alexandra typed in E fourteen times in succession. She had stored the graphics for ears in her Mouse program under the letter name E. The Turtle outlined the ears on top of the first set fourteen times. When she was asked if she liked seeing that, she nodded yes. Students repeated actions which pleased them. After Billy completed his first procedure, he felt
successful. He said, "Let's play that again. I think I want to play that again."

Sometimes students wanted to share their accomplishments with others. They wanted their kindergarten teacher to see their computer programs and graphics. They were also eager to demonstrate their programs to their kindergarten class. Several class presentations were made. The students typed in the names of their programs and explained how they worked. They identified the parts of the computer and explained their functions.

Billy liked to pretend that he was the Turtle. Masking tape was placed on the floor in the shape of a square. He enjoyed walking on the square in the Turtle "game."

**R** Well, we have some time, if you want to play DRAW or GAME. We have about five minutes left.

**B** Well, we've still got the square (on the floor). We can do that again.

**R** Would you like to go around the square again?

**B** Yes. You can count how many numbers.

**R** Okay. I'll count how many steps. You be the Turtle. Which way are you gonna go first, Turtle?

**B** This.

**R** Okay. Which way is that?

**B** Uh, Forward.

**R** Okay. I'll count your steps. 1, 2, 3, . . . 12.

**B** Twelve steps.

**R** Twelve steps forward.

**B** Pretend to push the button. [refers to typing the command]

**R** Okay. I'll pretend. F-D, space, 12.

Billy wanted to walk around the square as the Turtle again, even though he had done it two days before, because it was a pretend game that he enjoyed.
All of the participants except Billy engaged in randomly typing keys in no particular order. They liked to type fast and fill up the screen with characters until it could hold no more and the computer beeped. Jonathan said, "I want to just draw [type] something that doesn't spell anything.

Then he quickly typed groups of letters separated by spaces which had the appearance of words, but no meaning. Andrea also used the space bar in her random typing.

Students may have typed the keys randomly because they enjoyed the fast action and the immediate result of lines of letters and numbers. They may have been imitating adult typists they had seen. It is noteworthy that four of the subjects engaged in this kind of behavior without ever witnessing the others typing in such a way. Instances of random typing began during the first week of the study and continued through the programming posttests. When Andrea was working on the third dot task she said, "I wish I could get this finished so I could do my fun typing." Danielle was questioned about her random typing during the programming tests in the following excerpt.

D  It's fun doing that. (typing randomly)
R  Why is it so fun?
D  Because (pause) it's fun typing.

Students typed randomly more often at the end of the individual programming sessions when they were tired. However, some students interrupted their programming efforts with random typing breaks.
Students also enjoyed typing letters and numerals in order. When Andrea was asked for a number input for one of her programs, she responded in this way:

AN I know. (types 1234567890:)
R Why did you do that?
AN I just wanted to.
R Okay.
AN But I can still back up there, though.
(uses the delete key to eliminate the line of numerals)

Jonathan typed all of the keys on the keyboard going from left to right, pushing each one in turn. Alexandra typed in all of the letters in alphabetical order before writing her procedure for a square. She liked reading the error message: "I don't know how to ABCD..."

Several students typed in names on the computer. Some used their own names in special messages such as "Good work, Jonathan." Alexandra wrote a short program which printed "Hi Alexandra" and then drew the ears of her mouse program. In addition to typing their own names, they typed the names of their brothers, sisters, and friends.

Students occasionally sang or made sounds as they typed. Danielle was pleased with her rate of typing and said as she typed, "Yeah. I'm going quick, quick, quick. Now, now, now [sung at different pitches] and RETURN." Alexandra sang the alphabet song slowly as she typed the corresponding letters (with a few interruptions when she searched for a particular key).
Singing and sound making occurred spontaneously when students were busy typing and felt successful.

One other action which students chose to do was look at their reflections in the screen of the monitor when it was turned off. Andrea enjoyed making faces and used it like a mirror.

Students were allowed to select the preskill games they wanted to play. The two easiest games, DRAW and G, were the most popular. The one named GAME was seldom chosen. It required students to turn the Turtle toward the square on the screen and enter a number to make the Turtle travel toward the "box." In DRAW and G, students manipulated the Turtle by typing repeatedly F, B, R, or L. Billy mentioned that he liked G better than GAME as he was playing G.

Students were more successful when they played the easier preskill games.

The participants express preference for the keys which created punctuation marks. They liked the effects of the
REPT, RETURN, and the right arrow keys. Among their favorite commands were those which caused things to disappear from the screen such as HT or CS. Certain commands and numbers were used more frequently than others in programs simply because students "liked to."

Particular actions were repeated when students felt successful. Some participants asked to show their work to their teacher after they had completed programs. Students typed letters and numerals in sequences; they typed names which were important to them. The participants engaged in unexpected behavior such as random typing, singing, and playing with the reflections on the monitor. The preskill games, DRAW and G, were preferred to GAME. These responses to the computer expressed their emotions and individual preferences.

Program selection and names.—All students were asked to write a program for a house and to write three shape procedures at the end of the study. Students initiated the ideas for the other programs that were written. They chose their own names for their programs. A list of the program names is located in Appendix H.

There were few similarities among the kinds of programs created by the students. There were five programs which made squares and rectangles. Two students wrote programs for letters in their names. The students' final projects (on which they worked the longest period of time) created
graphics for: a mouse, a truck, the letter B, and a rose. Danielle wrote a program which spelled her name as her final project.

The names which students selected for their programs were either descriptive, or abbreviations of longer words, or both. (See Appendix H.) Most students chose descriptive names for their programs. These names were synonyms or contained adjectives which described the graphics. Jonathan named the procedure for the stem of his rose LOGO9 because it "looked like nine inches." He described the length of the line in the name he selected for his procedure. Some program names were abbreviated versions of longer names. Alexandra used more abbreviations as program names than the other students. She wanted to name her program for a mouse simply M. Andrea only used one abbreviation as a program name, and Danielle used very few abbreviations. Nine of the names for programs which students initiated were abbreviations of descriptive titles. Jonathan thought of several names of this type. He named his program for a rose BIGLONGFLOWER, but he abbreviated it as BGFR. (He selected the B in BIG, the final G in LONG, and the first and last letters in FLOWER as the initials in his abbreviation.)

Students also used descriptive words or abbreviations in the names of their house programs and their shape procedures. Table X compares the names of the procedures used in the house programs. Only three students' work is
represented because Billy was not asked to make a program for a house, and Andrea never finished her program. She said, "I got tired of the house." She lacked motivation to complete it because it was not her idea to make a program for a house.

**TABLE X**

COMPARISON OF PROGRAM NAMES FOR THE HOUSE TASK

<table>
<thead>
<tr>
<th>Students</th>
<th>Alexandra</th>
<th>Danielle</th>
<th>Jonathan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Program Name</td>
<td>H (House)*</td>
<td>DRAWH (Drawhouse)</td>
<td>SH (Spookhouse)</td>
</tr>
<tr>
<td>Base Procedure</td>
<td>SQ (Square)</td>
<td>S (Square)</td>
<td>SPOOK</td>
</tr>
<tr>
<td>Roof Procedure</td>
<td>TURTLE</td>
<td>TRI (Triangle)</td>
<td>PART</td>
</tr>
</tbody>
</table>

*Words in parentheses refer to the unabbreviated names.

Danielle began her program name with the word DRAW, which was her custom. Jonathan named his program Spookhouse because he was working on it before Halloween. Alexandra gave the roof procedure the descriptive name of TURTLE, when she was asked why she named it that, she replied, "Cause the top part looks like the Turtle, and the Turtle's a triangle."

The names which students selected for their shape procedures are presented in Table XI. Alexandra selected her usual one-letter abbreviations as shape procedure names, whereas Andrea insisted on spelling out the shape names in
The name of Billy's first procedure, B, was an abbreviation for the word BOX. He chose the phrase GOING AND STOPPING to describe the actions of the Turtle in the second shape procedure. Jonathan's procedure names were also very descriptive. BIGRECTANGLE and THREELINES both contained adjectives.

**TABLE XI**

**COMPARISON OF SHAPE PROCEDURE NAMES**

<table>
<thead>
<tr>
<th>Students</th>
<th>Procedure 1</th>
<th>Procedure 2</th>
<th>Procedure 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>S</td>
<td>R</td>
<td>T</td>
</tr>
<tr>
<td>AN</td>
<td>SQUARE</td>
<td>RECTANGLE</td>
<td>TRIANGLE</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>GOING AND STOPPING</td>
<td>TRI</td>
</tr>
<tr>
<td>D</td>
<td>SQUARE</td>
<td>RECTANGLE</td>
<td>TRI</td>
</tr>
<tr>
<td>J</td>
<td>BOX</td>
<td>BIGRECTANGLE</td>
<td>THREELINES</td>
</tr>
</tbody>
</table>

Students' names of procedures and programs reflected individual preferences in the use of abbreviations or complete spelling of the word and in the use of descriptive names. Students developed rather consistent practices in choosing names for their programs.

The kinds of programming projects initiated by the students varied. One of the research questions was: "What kinds of programming projects interest young children?"
Appendix H contains a complete list of the students' program names and a short description of the graphics they created.

Summary.—Students exhibited both positive and negative responses to the computer. They experienced positive feelings when their programming efforts were successful or created unusual results on the monitor. They expressed negative feelings when they were frustrated. On some occasions students talked to the computer or to the Turtle. Students' attention spans varied depending on external distractors and their physical and emotional condition.

Some of the participants expressed the desire to work independently and refused assistance. They demonstrated preferences for certain keys, commands, and numbers. They repeated actions which they enjoyed such as random typing and typing in letters and numerals in sequence. Students preferred DRAW and G to the more difficult preskill game, GAME. They expressed their individual creativity in selecting subjects for computer projects and in the names they gave their programs.

Thought Processes

One of the purposes of the study was to analyze the thought processes of the students as they worked with the computer. Their thinking was studied by examining their remarks, their actions, and their computer programs.
The thinking of the participants was revealed in the questions they asked and the experiments they tried as they used the computer. Their dialogue with the researcher contained some imaginative explanations of the operation of the computer. Students were becoming aware of words and sound-symbol relationships. They created original computer commands. Some of their explanations were more detailed and complete than others. Students devised their own aids to programming. Their individual programming styles and problem-solving strategies differed. They responded in a variety of ways to errors in their programs.

Questions.--Students asked the researcher many questions during the study. Most of the questions were in reference to the operation of the computer and to the words which appeared on the monitor. Students were curious about the restrictions on touching certain parts of the floppy disk. Andrea asked, "What happens when you touch it? Does it get ink on it? Jonathan wanted to know why holding it by the label was acceptable: "How come it won't ruin on this?"

Students asked about the consequences of certain actions: "What happens if it [floppy disk] bends?" Danielle wanted to know what the computer would do "if we pushed every letter in here."
Students had questions about the meanings of words they saw on the keyboard or on the monitor. Danielle pointed to the RESET key and asked, "What does this mean?" A frequent question "What's it say?" when students could not read long error messages such as NOT ENOUGH INPUTS TO LOAD. Sometimes they requested that commands be spelled.

The participants wondered how the computer performed certain functions: "How does it save it?" and "How is it erasing it? [referring to PENERASE]." They had questions about the operation of the computer. One girl asked, "Can we save it again?" As the computer was loading a program, Alexandra asked, "What's that noise?" Jonathan removed the floppy disk from the disk drive and wondered, "How come it's still on?" He also wanted to know why the computer had not shown his graphics when he was naming and typing in a procedure before he entered END. He was referring to the cursor when he asked, "Can you erase that right there?" Andrea wondered whether she could type in all of the commands for a shape procedure on the same line.

Students were curious about the operation of the computer and the meaning of the messages on the monitor. They asked about the floppy disk and the noises made by the disk drive. Appendix I lists other questions which the students asked about the computer and about programming.
Experiments.--Students experimented with the keys and the spacing of commands. They varied the size of the numbers they used as inputs. They tested several other original ideas on the computer.

Students used a trial and error approach in learning about the computer. Jonathan expressed this experimental attitude when he asked, "What if we did that and then did that?" as he typed two left brackets instead of a right and a left one. He also tried adding quotation marks before the number in a command: RT "26 and learned that it did not affect the turning of the Turtle. Andrea typed two hyphens after the number input: LT 90--. She discovered that they interfered with the performance of the command. She also tried pushing the RETURN key four times in succession and explained, "That's so I won't have to push RETURN [at the end of each line]."

Jonathan, Andrea, and Danielle experimented with adding extra spaces in their commands. After Jonathan typed two spaces instead of one, he said:

J What if I did it that many times?
R Let's try it and see.
J There.
R Did it work?
J Yep.
R Even with an extra space?
J (nods yes)

He also tried the opposite effect and omitted the necessary space between the direction and the number. He asked, "What if you didn't space that?" They learned that the computer
required at least one space in the proper place in the command, but that it would function as well with more spaces.

Students typed in nonstandard inputs to see what the computer would do. Danielle said, "Watch this. I'm gonna do somethin'. What if I typed CAT?" She was forming ideas about the inputs the computer accepted and those it rejected. Andrea tried the following experiment during the programming tests at the end of the study.

AN What would it do if I did this? (types B432S5G1)
R What do you think it would do?
AN I know what it will say: "I don't know how to B432S5G1."
R You want to find out?
AN (pushes RETURN; computer prints the message: "I don't know how to B432S5G1.")

Andrea's hypothesis was confirmed when she received the expected error message.

Students experimented with number inputs of extreme sizes. Jonathan typed RT 0 and asked, "What if you did that?" Andrea tried FD 0 and wondered, "Did it take a little, tiny step?" Students also tried very large numbers in their commands. Jonathan typed FD 678 and said, "I wanna see what that does. Goes three marks [wraps the screen three times]." He typed REPEAT 3360 [FD 1 RT 1] and made the Turtle draw a circle for a very long time.

One experiment resulted in a curve. When Andrea was typing in the preskill game DRAW, she typed F and L at the same time several times. She said, "He doesn't know which way to go." The Turtle went forward and left a little each
time she repeated the action. It drew a curved line on the
monitor and Andrea was pleased with the circle which resulted.
She exclaimed, "Oh, you know what that looks like? It looks
like an ice cream cone!"

Sometimes students experimented with the operation of
the computer itself. Danielle motioned to open the disk
drive while it was saving one of her programs, but the
researcher stopped her. She wanted to know what would happen
if she did that. Jonathan succeeded in finding out. He
typed LOAD "SG and then opened the disk drive.

J What if you opened the door like that?
R No, close it. (computer buzzes) What
happened, Jonathan?
J It made a noise.

Students did not hesitate to explore the functioning of the
computer. Danielle was asked to refrain from pushing the
ESC RESET key.

Some student experiments had beneficial outcomes. When
Andrea was typing in the commands for her truck program in
the EDIT mode, she suddenly stopped using RETURN after every
command and began typing them in across the screen, spacing
appropriately between each one. The computer accepted them
equally well that way, and the experiment saved Andrea the
effort of pushing the RETURN key so often.

Danielle, Andrea, and Jonathan conducted a variety of
experiments as they worked with the computer. However,
Alexandra and Billy did not. They were the careful, precise
typists. A review of their transcriptions did not reveal any incidences of experimental behavior. Occasionally Billy typed in large number inputs in GAME, but he was dismayed when he saw the results. He felt out of control when the Turtle wrapped the screen. When he played the preskill games, he used L exclusively to turn the Turtle for the first four weeks of the study. Then he tried using R as well. Billy was not a great risk taker.

Alexandra's behavior was on task except for the times she indulged in favorite actions such as random typing. She also created lines of numerals and letters with the REPT key. She knew what would be the outcomes of such actions; therefore, they were not truly experimental.

The trial and error approach of the other three students yielded useful information about the acceptable uses of punctuation marks and spacing in LOGO. Students also discovered some things about programming and the operation of the computer through experimentation.

Imaginative thinking.—During the study there were instances in which students attributed human characteristics or emotions to the computer and to the floppy disk. One incident of this nature occurred as Alexandra prepared to leave the room after a programming session. She pushed in both chairs and switched off the light. As she stood by the door and looked at the Turtle glowing on the monitor she
remarked, "The computer will think it's in a different room because the chairs are pushed in and the lights are out." She reasoned that the computer would be confused since both of those conditions were not normal.

Sometimes students attributed feelings to the computer. As Andrea walked toward the computer room with the researcher, she said that the computer would be glad to see us. Andrea added, "It is lonely by itself." She asked if the researcher took it home. Billy indicated that he thought the floppy disk had feelings. When he was asked to return the floppy disk to its sleeve at the end of a session, he said, "That's where it wants to sleep."

The disappearance of things fascinated the students. CLEARSCREEN (CS) was one of their favorite commands. Alexandra developed a theory to explain where the graphics went when they disappeared from the monitor.

A  C-S. It disappeared. It's probably in there somewhere [inside the monitor] hiding.
R  You think so?
A  It's probably right in there. That little square [her graphics].

Billy also talked about things hiding. After he put the floppy disk into the disk drive he said, "I let that disk hide. Nobody will find it." Jonathan was interested in the operation of the disk drive because its functioning was hidden in the box. He asked the researcher how the drive saved programs on the floppy disk.
R It's saving the program; it's writing SPOOK down on the disk so we can see it again tomorrow.

J How is it writing?

He wanted to know how the drive worked inside. He said, "I like about it when you put the disk in. I wish I could be one. I'd be in there looking at that little thing right there, the little circle [the center of the floppy disk]." Students were not allowed to touch the circular part of the floppy disk. Jonathan speculated, "maybe that's sharp and you can cut your fingers on it, if you press down real hard. . . . The thing in there [inside the drive] knows what it feels like." Jonathan used imaginative thinking to form hypotheses about what he could not touch or see for himself.

Students were curious about the operating sounds made by the disk drive when it was loading or saving their programs. Billy said, "It sounds like there's something in there. It sounds like 'tic, tic.'" He wondered what was causing the sounds and reasoned that it had to be something inside the drive that he could not see. Alexandra compared the sound of the drive to the sound made by an animal. She remarked, "It's noisy, isn't it? It's like a horse."

Although students used their imaginations in thinking about the computer, they knew that it was not alive; therefore, their thinking cannot be described as truly animistic. When she was asked at the end of the study, "What do you think the computer is?" Alexandra replied, "A machine."
Billy knew the difference between a real turtle and the LOGO Turtle. He made the distinction verbally on the second day of the study while he was playing G.

R Can you show me where the head is?
B This is the head of the Turtle right there. It's not a real turtle.
R No.
B It's not a real turtle. It's not the kind of turtle that lives in the zoo.

Students enjoyed engaging in imaginative thinking about the functioning of the computer even though they realized it was not alive. Alexandra had an idea for a program named COMPUTER that she thought would do something special.

A (types TO COMPUTER)
R What do you want it to do?
A It might show itself.
R You want the computer to show itself?
A On the screen it might.
R What do you think we would see there if we typed COMPUTER?
A The mouse.
R What did it do?
A Nothing.

She identified her program for a picture of a mouse with the computer itself. Alexandra thought she could make the mouse appear on the monitor by typing in the computer's name. She tested her imaginative idea and was disappointed when nothing happened.

Billy also created imaginative stories about the functioning of the computer. In the following incident the story served as a distraction from a programming test which he had failed.
B I just don't feel like doing a square.
R I'll just put the Turtle back where it was.
B I told you I don't feel like doing it.
I'll see what this is (pushes right arrow key)
Again. Must be the beep in your car. You
must bring your horn from your car in here and
it did that.
R You think that's how it works?
B Yes.
R I think you're making up stories.
B That's just on cartoons.

Billy's last statement indicates that he was able to distinguish fact from fantasy. However, that capability did not prevent him from thinking of an imaginative explanation for the beeping sound made by the right arrow key.

Some students' remarks reflected beliefs that the computer felt emotions and that it could be confused by alterations in its setting. The hiding of the floppy disk in the drive and the noisy, mysterious functioning of the disk drive were topics which concerned several children. Although students indicated that they did not believe the computer was alive, imaginative explanations and hypotheses persisted. Such imaginative thinking is natural for five-year-old children. At this stage of development they engage in pretend play with their peers. Therefore it is not unusual that they fabricated stories about things they did not understand. What is interesting is that some of the students displayed imaginative thinking on many occasions whereas others did not. The instances of imaginative thinking were counted and the results were as follows: Billy gave seven examples of such thinking; Alexandra had four instances; Andrea and
Jonathan each had one; and Danielle had none. There was a relationship between the frequency of the students' attributing human characteristics to the computer and the level of their performance on the Piagetian tasks. Danielle performed in the third stage of concrete operations on most of the Piagetian tasks. The absence of imaginative thinking was a sign of her maturity. Billy had more Stage I (preoperational) responses to the Piagetian tasks than did the other students. He also expressed imaginative thoughts more frequently than the other students.

Children who were developmentally younger engaged in imaginative thinking more often than children who were able to conserve on the Piagetian tasks. The frequency of the students' implausible, imaginative explanations was another indicator of their developmental maturity.

Print awareness.—The students demonstrated an awareness of the words on the monitor and those in the dribble file of commands kept in the researcher's notebook. Some students were beginning to form relationships between the symbols in print and the sound of the word or command. Students varied in their reading ability. New commands were invented and tried on the computer during the study.

All of the participants realized that print had meaning although some of them were unaware that the letters had to be written in a particular order to be interpreted. When
Danielle was typing lines of letters randomly she asked, "What does that say? What does that mean?" She expected the researcher to be able to read it. Andrea knew that CS cleared the screen. She asked the researcher what SC meant. She expected that combination to have meaning as well.

During the programming tests Danielle suggested that the researcher write "too little" next to a number which did not make the Turtle go far enough. Andrea was aware that there was a spelling difference between word commands and their abbreviations. She typed BK and then she said,

AN I want to do the long part.
R Okay. Do you remember how to spell BACK?
B-A-C-K.

She sometimes preferred to type in the full command rather than the abbreviation.

The boys in the study confused the meaning of letter and word. Billy was asked to type CATALOG so that he could locate the name of his program on the listing. He watched the researcher spell CATALOG in writing and remarked, "That's a big letter." As Jonathan typed PU (PENUP) for the first time, he commented "U is a new word." When Billy looked at the screen full of commands while the disk drive was loading his program, he said,

B That's a long letter down.
R Yes. Those are all the commands that the person before you put in the computer.

Sometimes students expressed frustration with typing lengthy commands. When Billy was asked to type LOAD "DRAW, he
refused, saying, "I think that's a long letter, so, uh, I can't think about anything else." He was too tired to attempt to type it. Jonathan said, "That sure is a big thing" when he was asked to type CATALOG. Alexandra had a similar comment about the word ERASEFILE. She said, "That sure is a long word."

Some students tried to "sound out" words during the study. They knew there was a relationship between letters and sounds. When Danielle wanted to name her program for a house, HOME, she knew the first letter was H as she said "Huh." Alexandra attempted to type CATALOG without looking at the command spelling on the poster above the computer. (She already knew the spelling for cat.)

A  Cat.
R  It's just one word. Good. Log.
A  Luh, luh, uh, uh, guh.
R  Okay, you need an O in between there.
   L-O-G. Good.
A  I didn't even look at that (poster).
R  I know it! You're getting so smart---sounding out words.

Although Alexandra read fluently, spelling words was a more difficult skill. Jonathan was also aware of sound-symbol relationships. He transformed his typing errors into words.

J  (types TR) That's trite.
   (writes RD)
R  What's this word?
J  Ride. (changes it to RT)

Alexandra was the only student who misspelled words because she knew alternative spelling rules which created the same
Mistakes were made in reading certain error messages. Andrea typed: BACK BAT BCAK BAT. She "read" the error message as: "I don't know how to back up bat!" Danielle, however, not only read difficult error messages correctly, she comprehended them as well. When the computer printed: NOT ENOUGH INPUTS TO RT, Danielle read it and was asked what the message meant. She replied, "I forgot a number." She consulted the dribble file for the number input she used previously and found it. Andrea also succeeded in finding in the notes certain commands she had used before.

Some students experienced difficulty in recalling program names and reading them. Other students read the names of their programs consistently well. Jonathan was not always able to recall the names of his programs. He had trouble reading them from the catalog of his disk also. The following excerpt from the transcriptions illustrates the difficulty of the task.

R Can you read those names?
J Uh-uh.
R Yes, you can. Try.
J Uh, CAT, I mean SPOOK.
R No, SPOOK's down here. SPOOK starts with an S. This is CARROT.
J And BOX, BOX.
R BOXES.
J BOXES.
R This is just like this word except it has a number on the end.
J CARROTS11.
R That's right, CARROTS11 and then,
J um, CARROTS. (incorrect)
R SPOOK. Okay. Those are all your programs, Jonathan.

The other participants demonstrated the ability to read the names of their programs. Billy had no trouble reading the catalog of his disk:

B (reading) HELLO [name of program which initialized the disk]
R Yes. L (pause)
B LOGO. I.LOGO. uh. RECTANGLE.LOGO. PUZZLE.LOGO.
R Do those sound familiar?
B Yes.
R Whose programs are those?
B Uh, mine.

Students used what they knew about reading, letter sounds, and spelling to create original computer commands. Andrea created her own abbreviation for HIDETURTLE when she typed HTRL. (The computer accepted only HT.) She said the letter sounds as she typed the letters. After Alexandra failed to write a workable triangle procedure, she wanted to erase the mistakes from the screen. She typed: BK ALL. (She asked the researcher to spell ALL.) She expected her new command to perform the function of CS (CLEARSCREEN). Danielle typed GO BK 76. The extra word at the beginning did not interfere with the execution of BK 76. However, her next attempt, GO UP 76, did not work. Jonathan indicated his understanding of correct spacing when he wrote the commands for his shape procedures on paper. He used periods to mark the spaces in the commands: FD.901, FD.30, and RT.89.
Students were aware that words represented a message in written form. They expected all letter combinations to have meaning. The boys displayed confusion about the differences between letters and words and sometimes mislabeled each. (That kind of behavior is characteristic of five-year-old children.) Students sounded out commands as they typed them on the computer. Jonathan had difficulty reading the names of his programs, but the other participants read the catalogs of their disks well. Although error messages were more difficult to read because some of the words were unfamiliar, some students succeeded even in that. Students invented their own notation for spacing commands and created original computer commands.

**Explanations.**—After students thought of a programming project, they defined how they planned to proceed—which part they would make first, and so forth. They communicated their plans with varying degrees of success. Sometimes students used motions to explain what they planned to do. Their explanations were very limited at times and very precise and detailed at other times.

Students pointed and used hand motions to aid in their explanations. After Danielle had decided to make a program for the letter N, she traced the path she wanted the Turtle to take with her finger on the screen. The following
example of using motions was taken from the transcript of
the day on which she completed her first procedure.

R Now which direction are you going? Is that forward
or back?
D Forward.
R I can't figure it out when you're just pointing.
You have to tell me the word.

Danielle was stepping on top of an E which she had drawn on
newsprint. She had been asked to verbalize each command,
but in this case she was only pointing straight ahead.
Jonathan used motions in a less controlled manner. When the
researcher asked him if he knew the way the Turtle could
draw both house procedures at once, he replied, "Yeah, it
just goes [motions with his hands on the screen and makes
airplane noises]."

Sometimes students' explanations were very limited or
nonexistent. When Jonathan was asked why he selected RT 89
(instead of RT 90), he did not give a verbal response. (He
may have shrugged his shoulders.) On another occasion he
gave a very short explanation of an unusual program name for
the stem of a flower.

R TO SHORTLONG: S-G. That's an interesting name,
Jonathan. How did you decide on that?
J Well, I thought of it at my house.
R You did? What does it make you think of?
Why did you say SHORTLONG?
J Um, because I wanted to.

Jonathan had justified plans before with the phrase "I
thought of it at my house," when he could give no other
explanation. Some of Danielle's explanations were also
limited. After she had typed DRAW5 several times she was asked to describe the graphics.

D It did somethin'.
R Well, tell me about it.
D Well, it kinds looks like somethin' neat.

She labeled the picture as "neat," but she could not express a more detailed description of it at the time.

However, later that day during a presentation to her kindergarten class, Danielle gave an elaborated description of the same graphics.

D See how it's making that?
R Tell them how it made your EXP (Experiment), Danielle.
D By pushing four fives.
R Can you show them the fives in your design?
D There's one right here, and one right here, and one right there, and one right there.
R Is there anything else you want to tell them about the computer?
D (wanted to show them the floppy disk)
R It's all right to take it out.
D That's what makes it go. This is what saves the programs too. The floppy disk. But you can't touch that and that.
R That's right because it will do what to it?
D It will make it not look good. Not do stuff good.

Her program description was more detailed in the presentation. Her explanation of the floppy disk was also specific. Students gave relatively complete explanations during their individual programming sessions as well. Jonathan planned to make a rose as his final programming project. His picture on paper consisted of an outer circle with three concentric circles representing the petals of the flower. He described his plan for that part of his picture in this way: "We can
make one circle, and then go in and make one circle, then
go in and make one circle, and then go in and make one
circle." Before Billy started to dictate the commands for
his square procedure test, he had formed a definite plan.

R What would the Turtle need to do first?
B He would need to go up here, up here, then
down here, then down here.
R How do we tell the Turtle to go up?
B By pushing Forward and just make it turn.

Billy had visualized what he wanted the Turtle to do and he
knew the first command he needed.

Students' explanations ranged from inadequate to very
specific. Students who gave very detailed explanations also
gave incomplete ones at times. Piaget's observations (1974/
1976) of the limited explanations of young children and their
use of motions were corroborated in this study. Sometimes
the participants relied on motions to express their inten-
tions when they did not (or could not) verbalize them.

Programming aids.—Students relied on practices they
invented to make typing and programming easier for them.
They spelled commands repeatedly and referred to notes. They
devised original ways of estimating distances and traced the
path they projected for the Turtle with a finger on the
screen.

All of the participants spelled the letters of commands
out loud as they typed them. Some students spelled the word
completely before they began typing. Then they spelled the
individual letters as they typed them. Jonathan pronounced and spelled commands repeatedly until he had finished typing them.

```
J TO PART. PART: P-A-R. PART (searches for R)
R R-T.
R Return. HOME: H-O-M-E.
R E-N-D.
J E-N-D. E-N-D. (typing) E-N-D-D.
```

Although he said some of the letters twice while he was looking for them on the keyboard, he only typed them once.

Another typing aid which students utilized was the researcher's notes of the commands which the students had dictated. Before Billy typed in the command REPEAT he told the researcher, "Just keep the paper there so I can remember what it is." He depended on seeing the word in writing to help him spell the command correctly. Andrea consulted the dribble file when she needed to recall the number inputs she had used previously.

```
AN Need it to go left. What did I do it? What was that? Was it forty-six?
R I'll let you see your other answer. (She looks at the notes.)
```

Jonathan asked to see the picture he had drawn on paper before he completed the corresponding program. He wanted to recall the details so he could make his computer graphics look like his plan.

Alexandra thought of a helpful programming aid. She suggested that a strip of masking tape be placed on the
screen at the edge of the split screen. She wanted the tape there to help her judge the distance available for graphics on the lower part of the screen just above the section for commands.

Several students used the strategy of projecting the path of the Turtle by drawing a straight line with the tip of a finger from the Turtle across the screen to the goal. This aid was used often when students played the preskill game GAME. They pointed the Turtle toward the box on the screen. Then they traced a line from the top of the Turtle to the box to verify the angle before they entered a number.

Alexandra and Danielle estimated some distances by counting the number of times their fingernail touched in between the two points on the screen. They equated the steps taken by the careful progression of their fingernail to the very small steps taken by the Turtle. Danielle also measured distances on the chalkboard in this way when she tried to decide on the size of the input. On another occasion Danielle wrote the letters D and A on the chalkboard, leaving a space in between them. She was trying to decide how far apart those letters should be in her name program. She wrote as many numerals in sequence as she could fit between the two letters. She counted as she wrote. This strategy gave her the idea to leave nine steps of space between the letters D and A on the screen.
The programming aids devised by the students provided assistance with certain problems such as remembering the spelling of commands, recalling inputs used previously, discriminating the amount of space available for graphics on the screen, and figuring angle and distance estimates. The original ways in which students solved these problems provide clues to their cognitive needs. Their programming aids used a variety of sensory approaches—auditory, visual, and kinesthetic—to resolve programming difficulties. They applied concrete solutions to abstract problems.

Logical thinking.—Students demonstrated logical thinking in their programming efforts as they defined the problem. They understood cause-effect relations, made predictions, and tried alternatives.

Students were asked to limit their first programming project to a picture with straight lines. Three students chose to write procedures for shapes. Danielle, however, considered a broad range of possibilities before she decided on a project. She thought of some plans which were attractive to her and then determined whether they were composed of straight lines.

D There's PacMan. But PacMan doesn't have all corners.
R You wanted to make a D. Why did you decide not to do that?
D Well, there wasn't no corners. I know it. There. (draws a picture of her idea)
R What is that letter?
D E.
In her book, *Children's Minds*, Margaret Donaldson (1978) notes that in problem solving it is essential "to register those features which are relevant to the solution" (p. 107). Danielle compared the salient features of several pictures to the criterion of straight lines before she selected an appropriate project.

Students were also asked to draw a picture of a house and then write commands which would make the computer draw their house. All of the drawings of houses included a front door with a knob and one or more windows. Three included a chimney with circling smoke. Some students drew a house picture which had many pleasing details and then adapted their plan to fit what they knew how to do in LOGO. Donaldson (1978) emphasizes "the importance of achieving a more detailed representation of task structure" (p. 107). As she drew her house picture, Danielle verbalized the structure of the house task within the constraints of the LOGO environment in this way:

R And what are those things?
D Curtains. And they're gonna have a little flower on them--a design. They're not gonna make the flowers on my thing because I think it would be too hard. There. Each one of them are different flowers. And I might not do these.
R You might not do the fence there leading to the door?
D And I might not do the sidewalk. Cause that would be too hard. And I'm not gonna make the smoke (from the chimney). And I won't make the handles (on the doors). But I just thought of a idea. I'll just make little square doorknobs.
Danielle was considering how she could make certain details of the house in LOGO. She did not know how to achieve the spiralling effect of the smoke, so she eliminated it from the structure of the task. The round doorknobs were modified as small square shapes and were included in her mental plan. (She knew how to make squares but did not know how to make circles at that time.) The planning and revisions occurred before Danielle started the programming process. She was defining the problem of making the house. She was working out a detailed representation of the task. Danielle was thinking logically.

Students were aware that their actions had effects as they worked with the computer. They were able to ascertain the cause of errors and remediate them. The following quotation from one of Alexandra's sessions illustrates this process.

R If you want to see GOOD, what do you need to do?
A Type GOOD. Return.
R What does that say?
A (reading the error message) "I don't know how to GOOD."
R Why do you think it doesn't know how?
A Because I didn't load it.

Alexandra's program GOOD had to be loaded into the computer's workspace before it became operational.

Danielle was also able to think of logical solutions to her programming problems. When she typed BK 89 to make the left side of her D, the Turtle disappeared from the lower edge of the screen.
Danielle knew that the opposite of backward (the command that caused the problem) was forward. Her logical response solved the problem. She tried FD 1 first. When she still could not see the Turtle, she typed FD 51 which made it appear on the screen.

Jonathan thought of a logical explanation for the failure of an experiment. He had typed in a command for a circle with a very large input (3360) for REPEAT. This caused the Turtle to circle repeatedly. He tried taking the floppy disk out of the disk drive to solve the problem.

R  Did it make it stop?  No.
J  That's because it (drive) already readed it (floppy disk).

His explanation displayed a very good understanding of the operation of the computer.

Students also demonstrated logical thinking when they predicted the actions of the Turtle. Alexandra knew that the computer would cause the Turtle to wrap around the screen and make the design on the opposite side if she typed a large number input. The researcher cautioned her that she was near the bottom of the screen as she was programming her mouse.

R  Now you know that the screen stops about here because it has to have room for the words. It's called splitscreen.
A  Or his body will probably end up here (pointing to the top of the screen).
R  You're right. His body would wrap the screen and end up above his head. That would look funny, wouldn't it?

Alexandra was anticipating the results of her actions. She was able to achieve the effect she wanted by predicting the consequences of actions.

Jonathan also made a logical prediction based on his knowledge of opposite actions. He had filled the screen with lines made by a program (BOXES) which repeated his BOX program eighteen times. He wanted the Turtle to back up and erase the boxes off the screen. (This effect could have been achieved with the command PENERASE BOXES.) However, Jonathan had the following idea.

J  Can you write it backwards? And then I think it will go backwards. It will take out all those eighteen boxes.

R  You think if we typed in BOXES backward, it would take out eighteen boxes? That's a good idea, but the computer doesn't do that.

Jonathan was disappointed that his logical approach of spelling BOXES as SEXOB would not remove them.

Students developed systematic strategies when they were searching for particular program names on the disk catalog. Alexandra and Andrea read every name on the catalog starting from the top until they found the one they had most recently saved on the bottom line. After several experiences of reading all of the names, Alexandra reasoned inductively where to look for the most recent entry: "It's always at the bottom."
Danielle also exhibited inductive thinking when she decided which letters in her name needed to be revised. She edited all of the lower case letters in her name to the same size before she wrote her DANIELLE program.

R  We've changed the 70 to 40. We have checked (the size of) A and N. We haven't checked I yet. You want to change I?  
D  To 40.  
R  Okay. Can you think of any letter that we didn't try in your name?  
D  L and D.  

Danielle mentally eliminated all of the letters she had previously revised and identified the letters which remained.

Alexandra learned about sequencing as she programmed the computer. When she wrote her procedure for the house, she listed the door first (which she had programmed last) and the result on the screen was a door on top of an upside down house.

A  It did it upside down.  
R  It did do it upside down. What a funny house!  
A  And that's the top of it (pointing to the roof).  
R  What do you think we should do to change it?  
    Let's type EDIT "H.  
A  I just move it down here?  
R  Where do you want to move it?  
A  I'll move that up there and put the D right here and END (wants to list the D last). It did it (correctly).  

Alexandra revised the program independently. Her understanding of the importance of sequencing was also demonstrated in this episode as she programmed one of the mouse's eyes:

R  What do we want the Turtle to do so it will make the eye?  
A  Say DOT on the computer.
R  Yes, we have to tell it to DOT. But remember
what happened yesterday when we told it to DOT
and then we told it to put PENDOWN? What happened?
A  It didn't make the eye.
R  So what do we have to do first?
A  PENDOWN. (types PD) Return.

Alexandra learned through these experiences that the order in
which commands were typed made a difference in the result on
the screen.

Andrea discovered that even when the Turtle was hidden
from view, it still acted on commands. This intrigued her.
She typed the HIDETURTLE command and then LT 90 and said,
"It turned" even though she could not see it. This kind of
belief in the stability of action in response to a command
could be termed "state permanence." It resembles Piaget's
concept of object permanence when infants attain the cogni-
tive maturity to realize that objects missing from view still
exist and look for them. State permanence refers to the
child's understanding that actions are performed by the com-
puter in response to commands whether or not the monitor is
on or the Turtle is visible. Andrea typed randomly one day
when the computer was on and the monitor was turned off while
she answered the questions in the student interview at the
end of the study. After she had typed several seconds, she
pushed the delete key and REPT to erase the invisible
characters from the screen. Although she could not see the
results of her typing, she knew that the computer had relayed
the signals to the monitor and she merely had to turn it on
to see them. The lines of typing existed, from her perspective, because she had developed state permanence. Therefore she performed the deleting action which "ate up all the letters."

Alexandra also demonstrated an understanding of the existence of something she could not see. She had learned that the Turtle's body took up space. When the Turtle drew a line, the line began from the back of the Turtle. Therefore when the tip of the Turtle was touching the desired position on the screen the line was short of the goal by a few millimeters. Alexandra had learned about the space which existed under the body of the Turtle which was only visible when HIDETURTLE was typed. She predicted that she would have to advance the Turtle more when it looked as if she had completed her graphics.

R The mouse's body is finished, isn't it?
A There may be a little space there. (under the Turtle)
R There might be. You want to hide the Turtle and see?
A Yes. H-T.
R Is there? (Alexandra nods.) How many Turtle steps do you think that is?
A Three.
R All right. Let's try FD 3. You're a wonderful guesser!

Both Alexandra and Andrea exhibited logical thinking in deducing the existence of actions and spaces they could not see without hiding the Turtle.

Students displayed a number of logical thinking skills during the study. They isolated the features of a problem
which were relevant to its solution. They explored cause-effect relations and made logical predictions. As they programmed the computer they developed systematic strategies and learned the effects of sequencing commands. Students exhibited inductive reasoning as they formed generalizations about the operation of the computer. They also made deductive conclusions about the existence or actions of things which they could not see.

Debugging.—Debugging is a computer term which refers to the process of finding errors and correcting them. During the study the participants demonstrated the ability to debug errors in computer commands and to locate bugs (errors) in their programs and fix them.

Students usually located spacing and spelling errors quickly. When Alexandra typed FD6, she remarked:

A  It says it doesn't know how to FD6.
R  Why not?
A  Because I didn't put a space.

Jonathan noticed the reason why the computer did not execute LOG009: "Cause I put two Os."

Sometimes students forgot to type part of the command and an error message resulted. When the computer printed, "I don't know what to do with 12," Jonathan diagnosed the problem: "Oh, I didn't put nothing in. I just put 12 in." He had forgotten to type in the directional part of the command. Students frequently left out number inputs in their hurry to type commands.
R What does it say?
A (reading error message) "Not enough inputs to FD."
R What does that mean?
A It can't go forward.
R Why?
A Because I didn't put the number.

Alexandra interpreted the error message and corrected the problem.

Students were also able to debug errors in the EDIT mode which displayed all of the program commands on the screen. They searched through many commands to find the mistakes.

R What a long program! Can you find out where we left out the number?
AN Yeah. L-T.
R Show me. That's correct. Now, type in 90.

Students were usually able to debug incorrectly typed commands without assistance.

The participants demonstrated their understanding of the relationship between certain commands and actions during the study. Because they recognized the effects of the commands, they were able to debug programs successfully. When Andrea was asked "What part of the drawing does REPEAT 360 [FD .5 RT 1] make?" she replied, "Oh, the wheel." Danielle illustrated her knowledge of the effects of commands when she edited DRAWL so that it would be the same size as the other letters in her name program.

D Ha, ha. Too big. (referring to the size of the L)
R Can you find the thing that you want to change?
The Forwards.

That's exactly right.

She knew that reducing the size of the forward inputs would make the L smaller.

Alexandra was able to debug a computer program without seeing the resulting errors on the screen. She did this twice during the programming tests. She found errors in two of her shape procedures simply by proofreading the written commands. In her procedure for a square, Alexandra changed FD 60 to FD 80 to match the other FD 80 inputs which she had written. In the rectangle procedure she changed LT 90 to RT 90 to match the other RT turns in the procedure. All of the other participants were able to debug their shape procedures after typing the commands and seeing the results on the monitor. Alexandra, however, was the only student who was able to debug a written procedure on paper.

Students were able to find many of their own errors independently. They located spacing and spelling errors. They were familiar with the structure of commands and noticed when any parts had been omitted. Students were capable of debugging their procedures. They located the errors and corrected them. One student was able to debug her procedure from the paper copy.

Problem-solving strategies.--Students utilized a variety of problem-solving strategies during the study. They tried to solve problems using approaches that had worked before.
Alternative methods were tried. They used opposite commands to correct their errors. Number inputs were changed systematically and additive strategies were employed.

During the programming tests Jonathan tried a number input that he had used before with success to move the Turtle to the dot.

J What was that one [number] that I did yesterday? Was it 90 or 60? Yesterday when I got to the dot?
R I don't remember, but the dots aren't all the same distance away so it might not work today even if it worked yesterday.
J Oh. I wanna see if it will do it again. (types FD 60)

He was determined to use the same input that had worked in the past.

Some students tried alternatives when the first effort failed to solve the problem. Danielle tried to move the graphics for the letter D to the left of the screen when she was programming her name.

R So how are we gonna move this D from the middle of the screen over here?
D I don't know. I know! First we have to put the stickers. What hand is this one?
R That's right.
D Okay. Now, you gotta go that way, right? (points to the left) So I gonna put LT "DRAWD. There. What does that say?
R (reading error message) "LT DOESN'T LIKE DRAWD AS INPUT."
D Oh, brother. What in the world are we gonna do?
R Have any other ideas?
D Only if we told it to . . . take how many steps do you um, how about . . . what is that? (writes 250) Two-five-zero. (types LT 250)

Danielle thought LT 250 would move the D horizontally to the left 250 steps. By inserting the number, she tried to
quantify how far she wanted the D to go rather than just
typing LT "DRAWD as she did in her first attempt.

Alexandra also demonstrated the use of alternative
strategies in problem solving. She tried two approaches in
the fourth dot task. (In this task the dot was positioned
at a forty-five degree angle from the starting point.) At
first Alexandra turned the Turtle to the right, moved it
forward so that it was almost under the dot, then turned it
left and moved it forward close to the dot. (See Appendix E
for a diagram of the fourth dot task.) This effort was not
quite successful and did not provide any information about
her knowledge of angle inputs, so the researcher asked her
to move the Turtle from the starting point to the dot in a
straight line. Alexandra varied her strategy, given this
new set of constraints. She tried turning the Turtle at an
angle less than ninety degrees: RT 50. Her second approach
was an appropriate alternative and almost solved the problem.

All of the participants except Billy made use of oppo-
site commands to correct their errors. Andrea employed the
FD PENERASE BK PENDOWN sequence six times as she worked on
the dot tasks. When she moved the Turtle past the dot with
FD 43, she typed PE and BK 43 to erase the line she did not
want. The command FD enabled the Turtle to draw again and
she tried another input. Students corrected wrong turns with
the opposite command. Right turns which were countered by
left turns of the same degree returned the Turtle to its original position. Then students tried alternative angles.

Students used systematic strategies to select alternative number inputs. Alexandra tried numbers of opposite sizes and systematically narrowed the range of possible numbers as she made the roof for her house. She tried RT 90 and the following inputs in this order: 300, 160, 130, 100, 110. She returned the Turtle to its original state after each trial except RT 110, which was the angle she desired.

Danielle also adopted a systematic approach to trying various inputs for forward commands. In the third dot task, she tried six alternatives before she was satisfied with the result.

R You gave it 70 and that was too far, wasn't it? So what number do you want to try now?
D How about 65?
R Here's the dot. It went a little past it. (She tries 64.)
D Now I know the one. (types 63) Now this time I really do know it: FD 62. Return. I think this is it. I think it's just right! (pause) I think we'd better type it in again. It should have been 61.

Danielle demonstrated a very good understanding of number sequence in this episode. She reduced the alternatives systematically as she neared the dot. Billy also tried smaller number inputs as the Turtle got closer to the dot in the first dot task.

Some students were willing to use larger numbers than others in their programming. Danielle was able to estimate
distance accurately. During the dot tasks, her first move was often a sizeable number such as 70. Then she would move the Turtle with smaller inputs (for example FD 20, FD 10) to the dot.

Billy and Jonathan were more cautious in their approach. Billy preferred to type in a series of FD 30s or FD 10s. Jonathan also typed numbers repeatedly. When he was making the stem for his flower, he typed five FD 3s in succession. Jonathan used a cumulative strategy in solving the dot tasks. He typed one number input, and if it did not make the Turtle go far enough, he continued typing inputs until he was satisfied. On the fourth dot task he typed: RT 7, RT 8, RT 0, RT 3, and RT 8. In the second dot task he tried the following cumulative inputs: FD 4, FD 5, FD 8, FD 5, FD 1, FD 4, and FD 1. All of his posttests were lengthy because of the relatively small number inputs which he selected. However, the sum of the numbers he gave was always within four steps of the required total.

Danielle was the only participant who used addition as a problem-solving strategy in her programming. On the first dot task she typed the inputs FD 20, FD 20, and FD 10. Then she turned the Turtle with the command SETH 0 instead of RT 90. She suspected that the dots were equidistant so she asked,

D What does these make altogether--like adding? (referring to the sum of the three commands) It'd be fifty steps, do you think? (typed FD 50)
R  I really don't remember, Danielle.
D  Just right. Wait, let's hide the Turtle. Yeah.

She solved the problem by typing in the sum of the inputs that had worked the first time. During the second dot task, she used the additive strategy again. The dots represented a square. Danielle typed FD 10 and FD 20 to move the Turtle from the first dot to the second dot. To move the Turtle from the second to the third dot, she typed FD 30. When the researcher asked, "How did you know to give it thirty?"

Danielle explained:

D  Because I put 10 and 20 and that makes 30.
R  Did you add 10 and 20 in your head? How did you know that was 30?
D  I just know.

She used the same additive strategy on the fourth dot task when she added fifty plus ten.

The participants employed a number of problem-solving strategies in their programming. Some of the strategies were indicative of the degree of risk the participant was willing to take. There were conservative strategies such as relying on methods which had worked previously and inching the Turtle along with a series of small number inputs. Students reduced the number of appropriate input alternatives systematically. Most relied on opposite commands to reposition the Turtle before they tried other inputs. Some strategies were more daring: moving the Turtle a great distance first and then closing the gap between it and the goal. Some students demonstrated an excellent understanding of number sequence.
and number combinations and used that knowledge to solve programming problems.

Responses to errors.—When students made programming errors, they either denied responsibility for the errors or they recognized that the errors were caused by commands they had typed. In the context of this section, errors refer to graphics results which were unexpected and differed from the child's plan.

During his interview with students, Piaget (1974/1976) observed that some young children did not seem to notice existing errors. Students in this study did not display that kind of behavior. They indicated that they were aware of programming errors. Their responses to the errors, however, were different.

Billy was the only participant who denied responsibility for some of his programming errors. He attributed the cause of the failure to the computer or to the researcher. When he was having difficulty with a programming test he said, "It's not my fault. It's the computer's fault." This is the same type of response given by some children who were interviewed by Piaget (1974/1980). Those children blamed mistakes on the pencil they were using in the task. Billy denied responsibility for errors on another occasion when he was proofreading his procedure for a square.
R The next thing you told it to do was RT 30.
B RT 30! What the heck dis I say that for?
R What should it be?
B It should have been RT 90. You did it wrong.
R You think I wrote it down wrong?
B Yes.

The error was made by Billy but he did not admit it.

More often, students did accept responsibility for their programming errors. Sometimes they did not attempt to correct the errors, however. When Alexandra's procedure for a triangle did not make a triangle on the screen, she recognized the problem but she did not try to change it.

R What do you think of the picture?
A It doesn't look like a triangle.
R Do you know why it doesn't?
A Why?
R If you were going to change something about it what would you change?
A Numbers.
R Which numbers? Would you change all the numbers?
A Except for 90.
R You would keep RT 90? Okay.

Alexandra knew that changing number inputs altered the appearance of a program, but she did not try her proposed changes. When Billy wrote his procedure for a triangle, he became very frustrated when he saw it was not working.

B Silly thing. We have to turn it by pushing RT 90. I wonder if this is the way I make a triangle. I don't think we'll make one, but if we don't, we'll just forget it.
R Are there some changes you can think of?
B I think that wasn't very good how to make a triangle.
R Can you tell me which numbers you would have to change?
B I just can't do that. I said I was gonna forget it if we just can't do it. I think I wanna go back to my class.
In this case, Billy recognized the error but he was not willing to make any revisions. He wanted to leave the situation instead.

Another response which students gave to unexpected results on the screen was acceptance. Sometimes the error had fortuitous results which the students liked and incorporated into their plan. Jonathan planned to draw a rose which was composed of concentric rings representing the petals. When he ended up with a different effect, two small contiguous circles within a large circle, he decided the mistake looked like the wings of a bee and retained it in his graphics. (See BGFR in Appendix N.)

The third response given by students when they accepted responsibility for programming errors was an attempt to edit the program and eliminate the errors. Danielle's first procedure, E, was turned on its side when it was typed the first time. She wanted to know, "Is there any way we can turn it like that?" (in an upright position). She searched for the error and edited the procedure until it looked like her picture of an E.

Students responded to programming errors in various ways. Billy attributed some programming errors to the computer or to the researcher. The other students knew that they caused the errors, but they differed in their reactions. Some did nothing to correct the errors. Others liked the
unexpected results on the screen and incorporated them into their plans. Students also worked to debug their programs and make the graphics conform to their plan.

Summary.—The thinking of the students as they were engaged in programming was evident in their questions and explanations. Sometimes they attributed human feelings to the computer and created imaginative stories to explain its operation. However they exhibited logical systematic thinking as well. They conducted original experiments to determine the limits of what the computer would accept. Programming aids were developed by students to help them remember commands and to simplify the selection of number inputs. Students were aware of the function of print and attempted to interpret error messages. They were able to find errors in their programs and correct them. The participants used a variety of problem-solving strategies. Some of their strategies demonstrated understanding of the sequence of numerals and of opposite pairs.

Students had different responses to programming errors. Some students recognized a problem existed, but they did not try to correct it. A few liked the unexpected results and adapted their plans to include the mistakes. One boy tried to blame the computer or others for his programming mistakes. According to Piaget (1974/1980), the refusal to accept responsibility is a less mature response than the acceptance
of the negation (error). Most students edited their programs until their graphics matched their plans. They accepted responsibility for their mistakes and corrected them.

Discussion of the Analysis of the Field Notes

A review of Chapter IV reveals several examples of Piaget's conception of success and understanding in the achievement of a task. Piaget (1974/1978) distinguishes between the achievement of a goal by trial-and-error (success) and achieving it through deliberate, systematic planning (understanding). Sometimes children accomplish their goals without understanding how they did so; they are successful, but they do not realize why their actions worked to solve the problem.

One of the students' problem-solving strategies in the study illustrates the achievement of success without understanding. Some students entered a series of small numbers in order to control the movement of the Turtle little by little. They continued entering small inputs until the Turtle had reached the goal. Students had favorite numbers which they preferred typing, but the numbers bore no relationship to the distance requirements of the programming problem. When Andrea typed FD 5 or Jonathan typed FD 3 successively, they did so because they wanted to—not for any logical reason.
On the other hand, there were instances during the study in which the participants demonstrated their understanding of how they achieved their goals. When they edited their programs, students selected certain commands which needed revision. They did not edit commands randomly. They realized the effect of particular commands and they could locate mistakes in spelling and spacing commands. Alexandra demonstrated her understanding of the effects of commands when she corrected her procedure after proofreading it on paper. In revising their programs, students corrected their work successfully and they understood the process.

Students' logical thinking strategies also exemplified Piaget's concept of success with understanding. Danielle isolated the relevant features which she wanted to include in her graphics as she defined the problem. She eliminated arcs and circles from her mental plan and devised a programming project which she knew she could complete successfully.

Some of the students' problem-solving approaches were very systematic. They used their knowledge of numeral sequence to make reasonable selections for inputs. Alexandra tried the following numbers: 90, 160, 130, and 100 during one programming session. She deliberately narrowed the selection of alternative number inputs. Alexandra achieved success in choosing an appropriate input because of her conceptual understanding of number quantity.
Some participants achieved success without understanding; that is, they worked out a procedure which was satisfactory but they did not really understand how it worked. Andrea's guess of RT 764 oriented the Turtle at a forty-five degree angle, but she could not explain why it did so. Other students understood how their programming efforts achieved successful results. Their use of systematic strategies, logical thinking, and advance planning enabled them to be successful programmers.

Case Studies of Student Programming

A detailed analysis of the students' computer programs was conducted to obtain specific information about the kinds of programming projects they selected. Details about the structure and style of programming were noted as well as the use of special commands such as REPEAT and DOT.

The students' computer programs were analyzed in conjunction with the corresponding transcriptions and field notes. The data pertaining to each child were sorted separately and examined before the individual case studies were written. Each of the following five case studies is divided into three sections: (1) computer programs, (2) programming styles, and (3) attitudes about programming.
Alexandra

Computer programs.—Alexandra wrote a total of twenty-one procedures. (Appendix J lists the commands and graphics for each of Alexandra's procedures in the order in which she wrote them.) Her first computer programs were geometrical shapes and designs: STEPS, PIN, and ANGLE1. STEPS was composed of three boxes of differing sizes: SHAPE, SHAPE1, and SHAPE0. PIN repeated the ANGLE1 procedure six times.

Although some students drew pictures on the playground of the way they wanted their computer graphics to look, Alexandra did not. She was able to make accurate number estimates without relying on counting steps outside. She did all of her work at the monitor.

Her house program contained three subprocedures: SQ, TURTLE, and D. BOX was worked out before SQ and the same number inputs were used to make SQ using the REPEAT command. The door of the house included a doorknob made with a DOT command.

Her final project was a lengthy program for a mouse (M). It consisted of seven subprocedures: C, DOTS, LINE, S, LONG, SH, and E. DOT commands were used to make the mouse's eyes. She worked on the M program for twelve consecutive programming sessions. Alexandra wrote two procedures which printed messages. COM printed "HI ALEXANDRA" and GOOD printed "GOOD BY ALEXANDRA."
Programming style.—Alexandra's programming was structured and organized. Her program for STEPS developed from her first procedure SHAPE. She decided to add a larger and smaller "shape" on either side of the first. She named the larger addition SHAPEl and the smaller one SHAPEØ. Perhaps the numbers in the names indicated their relative sizes.

Alexandra worked on one part of her picture at a time, named it, saved it and then combined all the parts (sub-procedures) into one program which executed the complete picture. This approach was encouraged by the researcher. It seemed to make sense to Alexandra whose pictures had several component parts. They could be programmed separately as subprocedures. Three of her programs contained subprocedures: STEPS, H, and M.

Attitude toward programming.—Alexandra was very interested in completing her programming projects. She began programming during the second week of the study and did not play the preskill games thereafter. Alexandra was able to concentrate on programming during the entire session. Her behavior was reserved and nearly always on task. She learned programming commands quickly and required very little assistance by the end of the study. Alexandra preferred to work independently; she often typed commands silently for several minutes at a time. She was systematic in her typing habits.
After she typed a command, she checked the monitor before she pushed RETURN.

Andrea

Computer programs.—Andrea's programming consisted of five separate procedures. (See Appendix K for the computer printout of Andrea's procedures.) Her first procedure, RECTANGLE, looked like a rectangle. However, the forward inputs were all different. The unequal sides resulted from the fact that she drew a rectangle on newsprint and stepped off each side. The sizes of her steps differed or the picture was not accurately drawn because the sides of the rectangle were FD 11, FD 7, FD 9, and FD 6.

CONE, her second procedure, was created to represent the base of a snowcone. The top part of the snowcone was made by typing F and L in the preskill game DRAW. It could not be saved.

Andrea's third procedure, W, represented a window. In her plan for the procedure, she drew two windows on the chalkboard: one of the windows was open and the other was closed. She wrote "MAMA" next to the first window. Then Andrea told this story: "My mommy told me to put the window down and then I put the window down. See?"

Her fourth procedure, PART, was the roof for the required house program. She completed procedures for the room, the base, and a door for the house in three days.
She saved the PART procedure. On the fourth day she did not want to type and save the other two procedures. Andrea said, "I got tired of the house."

She did finish difficult tasks in which she was interested such as her final project, SMOKY. Andrea completed SMOKY in eight programming sessions. SMOKY was a picture of her father's truck. She used the REPEAT command to make the wheels on the truck.

**Programming style.**—Andrea wrote two short procedures and two long ones (W and SMOKY). W and SMOKY were written as continuous lists of commands; they were not divided into subprocedures. One unusual method which Andrea employed was typing five lines of commands, each having more than one command, near the end of the SMOKY program. None of Andrea's programs was divided into subprocedures.

**Attitude toward programming.**—Andrea enjoyed working with the computer. She often laughed and thought things were funny. She was enthusiastic about playing the preskill games. Although she began programming during the second week of the study, she continued to play the preskill games into the fourth week of the study. She asked to play G once again during the sixth week of the study.

Andrea was distractible and off task frequently. She was talkative and asked many questions. Andrea enjoyed
experimenting with the keys and typing randomly. At times programming seemed less important to her than experimenting.

Billy

Computer programs.—Billy wrote six separate procedures. They were all straight-sided letters or geometric shapes with ninety degree angles. The commands and graphics for Billy's procedures are provided in Appendix L. His first and third procedures, L and I, were originally drawn on the playground with chalk. The others were drawn only on the monitor.

Billy did procedures he could finish quickly. Three of his procedures (L, I, and PUZ) were finished in one day each; the other three (RECl, REC2, and B) required two days to complete each of them. His procedure for the letter I was merely a straight line. REC1 was really a square. PUZ stood for the word puzzle. It utilized a REPEAT command to create four contiguous squares.

Billy was not asked to write a program for a house. That project was too difficult for him in terms of the angles required for the roof, the combination of subprocedures, and the length of time required to complete it.

The procedure B represented the first letter of Billy's name. He attempted a procedure for the letter Y one day but never completed it. He thought it was difficult and did not resume work on it even though he had finished half of it.
Programming style.--Billy wrote six short, simple procedures. He did not divide any of them into subprocedures.

Attitude toward programming.--Billy enjoyed playing the DRAW and G preskill games. He said, "It's in that box!" when he accomplished the goal in G. Billy rarely spent all of a session programming. He chose to play the preskill games for a few minutes nearly every day. According to Billy, programming was "hard to do" but the preskill game G was easy. Sometimes he interrupted programming with statements like, "I just want the Turtle in the Box." Then he played G the rest of the session.

Billy often expressed frustration while programming. He was very conscious of time passing during the sessions with the computer. He reminded the researcher to set the timer. (He knew that his session was over when it went off.) Billy made comments such as, "I think we're almost running out of time." These comments signaled his frustration with the task and an eagerness to return to his kindergarten class.

Danielle

Computer programs.--Danielle completed fourteen procedures. Most of her procedures created graphics of letters or numbers. (Refer to Appendix M for a complete listing of her procedures in the order in which they were written.)
Her first procedure, E, was drawn on newsprint and stepped off indoors. DRAWA and DRAW4 were drawn with chalk on the playground. DRAW5 and her other procedures were worked out on the monitor. DRAW5, EXP (Experiment), and LOGO4 used REPEAT commands to create curves and repetitive designs.

Danielle's house (DRAWH) was very symmetrical. It consisted of two subprocedures: S and TRI.

Danielle decided to write a program for her name as her final project. She completed the program for her name in twelve days. DANIELLE was composed of six procedures for letters. They were sequenced in a lengthy final program which positioned and spaced each letter on the screen. She revised the letter programs she had done previously for E and A so that all the letters were approximately the same height.

Her D program used the REPEAT command to make the curve.

**Programming style.**--Danielle wrote the program for her name as a long continuous procedure. DANIELLE called up the procedures for the individual letters within the program, but it was not formally subdivided into subprocedures.

Danielle's house procedure was organized into separate subprocedures: S and TRI. It was the only one of her procedures which employed subprocedures.

**Attitude toward programming.**--Danielle liked to program the computer. She started her first program during the second week of the study and did not ask to play the preskill
games after that time. Danielle was eager to learn the commands and attempt new projects. She had a confident attitude. Danielle persisted in making changes in the program until she was satisfied with the graphics. She spent a week on her DRAW5 program making adjustments to the top so that it looked just right. She also worked a week on her DRAWD program getting the curve perfected. She was a facile programmer as well. (DRAWL was completed and saved in eleven minutes.) Danielle was task oriented and usually used all of the time during the session for programming.

Jonathan

Computer programs.--Jonathan wrote eleven procedures. His graphics and procedures are located in Appendix N. Jonathan's first two procedures were geometric shapes: CARROT (triangle) and BOX (square). He drew CARROT on newsprint inside and on the playground outside. He estimated the inputs for his other programs inside at the computer. The complete name of BOX was "The Turtle in the box." It resembled the position of the Turtle on the computer's screen. He used the REPEAT command to form his third and fourth procedures: CARROT11 and BOXES.

Jonathan made his house project in two sections: SPOOK and PART. His project was an outline of a house; it did not have a square base and a triangular roof like the houses made by Danielle and Alexandra. The subprocedures divided
it at the midpoint of the roof into left and right halves. SH stood for the word spookhouse because it was programmed before Halloween. The title of Jonathan's house reflected his interest in the holiday.

Jonathan's final programming project was to make the graphics for a rose. He completed it in nine programming sessions. He named it Big Long Flower and abbreviated that as BGFR. It consisted of three subprocedures: LOGO9, SG, and RTCIRCLE. Jonathan utilized five REPEAT commands in the rose program to make the curves of the leaf, the outline of the flower, and the bee inside.

**Programming style.**—Although Jonathan's programming appeared structured, he approached it as a continuous process. Two of his programs were composed of subprocedures: SH and BGFR. In his spookhouse program, Jonathan typed commands for about half the house and then he was asked to name and save what he had worked out. When he finished the second half of the house, he did likewise. Therefore the end result was two subprocedures in SH, but he did not plan them in advance. The same thing occurred in the rose program. The researcher periodically asked him to name and save what he had completed.

**Attitude toward programming.**—Jonathan was interested in programming. He started his first program in the third week of the study. He played the preskill game G for the
last time during the fourth week. Jonathan spent the sessions on programming and experimenting. He was quiet and gave short responses to questions. He asked some questions related to experimenting: "What if I tried . . .?" Jonathan needed a lot of spelling prompting and assistance throughout the study.

Discussion of the Case Studies

Computer Programs

Students varied in the number of procedures they programmed and saved on their disks. Alexandra had the most with twenty-one procedures; Danielle was next with fourteen; Jonathan had eleven; Billy finished six; and Andrea saved five. (Andrea wrote two more procedures for a house program which she never saved.)

Students first tried to program geometrical shapes such as rectangles and triangles. Billy and Danielle, however, wrote their first procedures for letters (which contained right angles). All students wrote some procedures for shapes which had angles greater than ninety degrees except Billy. All of his procedures turned with the commands RT 90 or LT 90. All students used the REPEAT command. Only Alexandra used the DOT command in her programs. She was also the only student who saved procedures which printed messages.

Most students spent many hours working on their final programming projects. They completed them in eight to twelve
sessions. The exception was Billy who never spent more than two sessions working on a programming project.

All of the students except Alexandra drew with chalk on the playground at least one picture they wanted to program. Then they measured the size by counting their steps as they followed the large-scale outline. Alexandra did not need that preparatory strategy in order to select numbers which were appropriate, so it was not recommended to her.

**Programming Style**

Seymour Papert (1978) noted two different programming styles among sixth graders in the Brookline LOGO Project. Students were either systematic and planned their programs in advance or they tinkered around and a program emerged from their manipulations of the Turtle.

The five-year-olds in this study combined aspects of those two styles into a unique approach to computer programming. This approach differed from the two programming styles observed by Papert in two respects: (1) the students in this study were too immature cognitively to plan and write subprocedures in advance as the sixth graders did, and (2) the five-year-olds were not true bricoleurs or tinkerers because they programmed with a specific objective in mind.

The participants were systematic in that they defined their programming project in advance by drawing a picture
of what they wanted the computer graphics to look like. They talked about which part of their picture they wanted to make first, second, and so forth. Their drawing served to mediate their conceptualization of the programming goal (Piaget, 1974/1976, p. 328). The students' style resembled "tinkering" as well because they typed in a continuous list of commands until the Turtle finished drawing the graphics the way they wanted.

Although student programs were structured in two ways, continuous and subprocedural, actually the predominant programming style was continuous. Students' continuous lists of commands were sometimes named and saved as subprocedures at the urging of the researcher.

The subprocedural structure met the time constraints of the situation. Programs which were saved in small segments could be loaded and drawn quickly on the screen. Then work could resume on the graphics. The alternative (as in Andrea's SMOKY program) was for the researcher to type in ahead of time the commands of the previous days so that the participant could spend the programming session completing the graphics. This was time consuming and not generally feasible.

**Attitude Toward Programming**

All of the students had positive attitudes toward programming except Billy. Andrea spent a lot of time
"experimenting" with the keys instead of programming.

Alexandra, Danielle, and Jonathan spent most of their sessions programming. These students preferred programming to playing the preskill games. They persevered in working on programming projects until they were completed to their satisfaction. Billy and Andrea diverted their attention from programming when they became frustrated or distracted. Instead of programming, they chose activities which were more enjoyable, such as playing the preskill games or typing randomly.

In his profiles of individual students' work in the Brookline LOGO Project, Dan Watt (1979) described the affective consequences of learning to program on a learning disabled sixth grader. The boy became more interested in all of his schoolwork and developed better peer relationships. Special modifications were made for the two students in the Brookline project who could not write procedures independently. LOGO programs were written for them which allowed them to change number variables in a procedure which made designs (POLY) and to type stories without defining them as procedures. They were able to experience success with the computer because of the alterations made to fit their individual learning needs.

It is evident from Billy's comments and behavior that programming was not a rewarding, fulfilling experience for him. He was not capable of programming independently and
often felt frustrated. Billy's success with the preskill games was responsible for his positive attitude toward the computer. He expressed great uneasiness and insecurity about programming, however.

Age-related developmental characteristics of children should be considered when examining the attitude of students toward programming. According to the Gesell Institute's description of typical five-year-old behavior, security is of utmost importance to the child. "He is very careful to attempt only what he can achieve. And he is a very good judge of his own abilities" (Bates, Gillespie, Haines, & Ilg, 1979, p. 31). Billy did not want to try some programming projects because he was not sure he could succeed ("It's too hard for me") and his security was threatened.

Summary

There appeared to be a relationship between the number of procedures completed by students and their attitude toward programming. The students who completed the most procedures, Alexandra, Danielle, and Jonathan, all liked programming. The students who completed the fewest procedures, Billy and Andrea, preferred playing preskill games or experimenting with the keys. They completed fewer procedures because they stopped working on projects before they were completed and they devoted less time during the sessions to programming projects. Alexandra, Danielle, and Jonathan
completed all of their programming tasks. They had positive feelings about programming because they were successful programmers.

Discussion of the Instruments, Field Notes, and Case Studies

A comparison of the students' performance on the instruments, their behavior and comments recorded in the field notes, and their computer programs yields some interesting findings. Although the relationship between the performance of the students on the programming tests and their performance on the Piagetian tasks remains unclear, there does not appear to be any relationship between their performance on the Piagetian tasks and the number of procedures they programmed during the study. Andrea gave seven Stage III responses and two Stage II responses on the Piagetian tasks, yet she wrote only five procedures during the ten-week study. Jonathan and Alexandra gave more Stage I (preoperational) and Stage II responses on Piagetian tasks than Andrea did. However they each completed more than twice as many procedures than Andrea. Both Andrea and Alexandra missed the same items on the programming tests, but Andrea saved only one-fourth the number of procedures that Alexandra did.

What accounts for the difference in the number of procedures completed by the participants in the study? Three factors may have contributed to the difference: (1) low task perseverance, (2) greater interest in activities other
than programming, and (3) reliance on favorite number inputs rather than logical thinking strategies. These factors also describe Billy's behavior and attitude. He worked out only six procedures during the study.

Both Andrea and Billy experienced considerable difficulty in focusing their attention on programming; their commitment to finishing programming projects was generally low. This trait appears to be unrelated to developmental maturity or to the students' performance on the programming tests. (Billy and Andrea had diverse scores on both of those instruments.) Andrea was distractible and preferred at times to experiment with the keyboard rather than to program. Her comparatively low task perseverance prevented her from devoting as much time to programming as the other students did. Billy, too, exhibited low task perseverance. His fear of failure inhibited his programming. When he felt insecure, he requested to play the preskill games or to return to his kindergarten room. He responded to errors in his procedures by blaming other people or the computer or by quitting. Billy debugged only one procedure during the study.

The second factor which may have contributed to the few procedures written by Andrea and Billy was their attitude toward programming. Their attitudes represented the extreme opposite poles along a continuum of risk taking. Andrea was fond of experimentation; Billy wanted security. Both children
preferred to engage in activities which would satisfy these desires instead of programming.

The third factor which both Billy and Andrea had in common was that they generally relied on using favorite number inputs instead of selecting numbers in relation to distance estimates. Compared to the other students, fewer instances of logical thinking were located in the transcriptions of sessions with Andrea and with Billy. In most situations in which they had completed procedures successfully, Andrea and Billy did not understand how they accomplished the result.

The three factors may have perpetuated the existence of each of the others. It is probable that Billy's and Andrea's reliance on favorite numbers was a slow, tedious approach (compared to logical strategies), which in turn worsened their attitudes toward programming, and induced them to do something other than programming tasks.

In contrast to Andrea and Billy, Danielle and Alexandra produced many more procedures because they were able to sustain their attention in programming. The longest measured attention span during programming was Danielle's thirty-five minute programming session. Alexandra maintained her attention in programming for twenty-five minutes. Danielle wrote fourteen procedures (one of which was very complicated) and Alexandra wrote twenty-one procedures. Although Andrea's
longest recorded attention span was twenty minutes, she continued to conduct experiments with the keys throughout the session. Jonathan sustained his attention for twenty minutes during his longest programming session; he wrote eleven procedures. Billy stayed interested for fourteen minutes during his longest programming session and wrote six procedures during the study. The maximum amount of time Billy devoted to completing a procedure was two days. It was not unusual for the other students to spend as many as five days on one procedure.

Andrea and Billy had relatively few procedures completed by the end of the study. Admittedly, Andrea's SMOKY program was lengthy and was the equivalent of several sub-procedures. However, the results of the instruments indicate that she may have been an "underachiever" in terms of her programming performance. Andrea scored well on the programming tests and her performance on the Piagetian tasks was advanced. Nevertheless, her distractibility and low task perseverance interfered with the number of procedures she produced during the study.

Although Billy's performance was the poorest of the participants on the Piagetian tasks, he was not necessarily immature for his age. Billy was probably much more advanced developmentally than some of his other kindergarten classmates. He was unusually conscious of time and knew the month and date of his birth—a behavior not typically displayed
until the age of six and one-half years, according to Ames et al. (1979, p. 51). Therefore his lack of programming success must be attributed primarily to his low task perseverance, which may have in turn been caused by his lack of logical thinking strategies.

Three factors were isolated which were common to the students who produced significantly fewer procedures than the other students. The factors were (1) low task perseverance, (2) greater interest in activities other than programming, and (3) reliance on illogical problem-solving strategies. The results of the instruments, the transcriptions, and the production of computer programs in the study indicate that these factors may explain the lack of programming success experienced by Billy and Andrea.
CHAPTER BIBLIOGRAPHY


CHAPTER V

SUMMARY, FINDINGS, IMPLICATIONS AND RECOMMENDATIONS

Chapter V is divided into four sections: first, a summary of the problem and the procedures of the study; second, a summary of the findings; third, implications for educational implementation; and fourth, recommendations for future research.

Summary

The problem of this study was to describe and analyze the computer programming abilities and thought processes of five-year-old children using a conventional microcomputer and the Apple LOGO language.

Six kindergarten children were randomly selected from a group of ten five-year-olds who passed a screening test of numeral and capital letter recognition. One of the boys moved away during the study leaving a remaining sample of three girls and two boys, all of whom were Caucasian. Each student met individually with the researcher for instruction in LOGO for twenty to thirty minutes daily during a period of ten weeks. The researcher had previously completed one college level course in Apple LOGO.
The instruments administered during the study included: a test of number quantity (part of the screening test), nine Piagetian tasks, a student interview, two types of programming tests, and a parent questionnaire. Data was also collected during each daily session by the researcher in a book of field notes. The field notes contained a list of all of the commands typed by the student during the session, comments about the child's behavior, and quotations of some of the remarks made during the session by the participant. Each session was tape recorded and transcribed. The students' procedures on their individual floppy disks were printed out.

The data were analyzed in three sections: instruments, field notes and transcriptions, and computer programs. The results of the instruments were analyzed and compared. The field notes of each session were combined with the transcription of the corresponding session. Duplicate copies were made. The field notes and transcriptions were analyzed into five major categories of student behavior with the computer: (1) conceptual difficulties, (2) perceptual difficulties, (3) physical difficulties, (4) affective responses, and (5) thought processes. The data in these categories were further divided into numerous subsections. The computer printouts of student procedures were analyzed in the case studies of student programming.
Findings

The findings of the study pertain only to the five children involved and cannot be generalized to a larger population. The participants' computer programming abilities varied. All of them worked out procedures on the computer with the assistance of the researcher. The programming tests indicated that two students were independent programmers, two students could write procedures, but they had difficulty correcting their errors, and one student could not be considered a programmer. He preferred playing the preskill games on the computer.

The number of procedures written by the students differed and indicated their overall interest and ability to maintain their attention during programming. Three factors were identified which may contribute to programming success: (1) task perseverance (the ability to continue working on a task until it is completed), (2) degree of interest in programming (as opposed to interest in easier activities), and (3) reliance on logical thinking strategies instead of repetitive behavior.

There was no clear relationship between the performance of the students on the Piagetian tasks and their programming competence as measured by the results of the programming tests and the number of procedures completed during the study.
Implications

The implications of the study for the educational setting parallel in part the results of the programming tests.

1. Some five-year-old children are capable programmers. They can define their own projects and program with a minimum of assistance. School districts developing computer literacy curriculum should take this into account.

2. Some five-year-old children enjoy programming, but they require some assistance in working out programming problems. The assistance they need could be provided by a team of teachers who use flexible instructional arrangements so that one teacher is stationed at the computer for part of the day. In a self-contained setting, volunteers could help students at the computer.

3. Some five-year-old children do not enjoy programming. Appropriate options should be made available which meet their developmental needs for security and success.

One possible approach is to provide especially designed computer programs which enable students to use the computer as a medium to represent their thoughts in a way that does not require a lot of typing. Two such programs were designed for students with special learning needs in the Brookline LOGO Project. Dan Watt (1979) explained that one of the special programs allowed a student to make interesting designs by varying the size and angle inputs. No commands
were typed, just the number inputs (p. 15.6). The other program enabled a student to type and save stories and letters without typing PRINT commands (p. 16.7). These special programming modifications allowed students who could not program in LOGO to have a successful and intellectually challenging experience with the computer.

Another option is to introduce five-year-olds to the prerequisite skills of programming in LOGO through preskill games. These games provide children with experiences orienting the Turtle in space, using it as a drawing instrument, and learning about number quantity. The computer is programmed to move the Turtle a certain number of steps when the keys F, B, R, or L are typed.

The third alternative for children who are not ready for the complexities of programming is to postpone programming until they are older. The computer may be used with such children to reinforce skills in computer-assisted instruction. Ruth Ault (1977) advises wisely:

Learning takes time. If we try to teach a child something too early, it will simply take more time than if we wait until he is ready. The difference in the two lengths of time is the time we cannot spend teaching him other things. Before devising ever more clever teaching gimmicks, we ought to consider the costs and benefits to the child of learning that particular skill at that particular time rather than at another time (p. 176).

Educators jeopardize the child's self-concept as well as his future attitude toward computing if programming is taught before the child is capable of doing it successfully.
The final implications are derived from the analysis of the field notes and the reports of the case studies.

4. Students should be allowed to define their own programming projects. Students sustain their attention longer when they have conceived and planned their own projects.

5. Educators should encourage the development of task perseverance and logical thinking strategies in every area of the curriculum. If students learn to accept responsibility for their errors, try systematic approaches, and complete their work, they will have developed some skills which will help them to be successful in computer programming.

Recommendations for Further Research

The following are recommendations for future studies of the programming capabilities of young children:

1. Similar studies should be conducted using a color monitor, peripheral devices such as the Turtle robot with a pen, and the version of LOGO which allows users to save commands directly from the screen without retyping them.

2. Similar studies should be conducted with a heterogeneous sample of children from various cultural and socioeconomic backgrounds.

3. The relationship between performance on the Piagetian seriation task and programming competence should
be further explored to determine whether the task is an indicator of the types of visual perception and organizational skills necessary for programming in LOGO.

4. Similar studies should be conducted with five-year-olds using the Gesell Behavior Tests instead of the Piagetian tasks.

5. Further research should explore the relationship between task perseverance and logical thinking strategies upon programming competence.

6. Studies should be conducted concerning the relationship between the child's self-esteem, the measure of the child's locus of control, and programming competence.

7. Additional research should describe the thinking and behavior of young children as they learn to program in pairs.

8. Studies are needed of the effects of verbalization on computer programming. The problem-solving strategies of the participants should be examined. Gagné and Smith (1962) reported that the practice of justifying actions verbally "facilitates the discovery of general principles and their employment in solving successive problems" (p. 18). Papert (1980, p. 197) also uses a "loud thinking" strategy in his work.

In conclusion, this study has demonstrated that some five-year-old children are capable of programming a conventional microcomputer. Students varied, however, in
their programming competence and in their commitment to completing programming projects. Educators who teach young children to program should approach the task with a great deal of patience, an understanding of the developmental needs of children, and a tolerance for off task behavior. They should also listen carefully for conceptual difficulties and provide assistance when needed with visual and auditory discrimination problems. Computers are tools for thinking, but they should be used appropriately so that the individual emotional needs of children are respected and their intellectual abilities are encouraged.


APPENDICES
Appendix A
Letter to Parents

September 13, 1982

Dear Parent,

I am conducting a study of five-year-old children's computer programming abilities at _______ Elementary School. I would like your child to participate in this study.

I would be working with your child fifteen minutes each day at the school. We will learn how to make the computer draw pictures on the display screen. This will be a fun activity and an educational experience for your child. Your child's kindergarten schedule will not be substantially disrupted. Each child in the study will be able to play outside with the other children and be with the group during group time.

I am an experienced teacher with a kindergarten endorsement. The information gained from the study will be reported in my doctoral dissertation. The study will begin this week and continue through November 19. You will be informed about your child's experiences with the computer at the end of the study.

Please answer the questions on the next page and sign the permission form if you would like your child to work with the computer. I will pick up the last page from your child's kindergarten teacher. Thank you very much.

Sincerely,

Sandra N. Hines

Sandra N. Hines
No   Yes

☐  ☐ There are circumstances which prevent my child from attending school regularly.

☐  ☐ We may move to another school before December.

☐  ☐ We have a computer at home.

☐  ☐ My child uses the computer at home.

I give permission for my child, ____________________, to participate in the LOGO computer programming study. I also give permission for photographs to be taken of my child during the study which may be used in presentations at professional meetings and in journal articles.

________________________
Parent's Signature
Appendix B
Screening Test

Name: ___________________________ Date: ________________

1. Tell me the names of these numbers:

5__  6__  25__  10__  20__
8__  4__  90__  50__  60__
1__  9__  45__  80__  30__
7__  3__  100__  40__  70__
2__  0__

2. Which is more? 7 or 2? 100 or 45? 25 or 90?

3. Hold up your right hand. __   Turn to the right. __
   Hold up your left hand. __   Turn to the left. __

4. Stand up. Move forward. __
   Move backward. __

5. Here are three shapes. Point to each one and tell me
   its name.   S__  C__  T__

6. Tell me the names of these letters:

   F__  C__  R__  K__
   B__  T__  L__  A__
   S__  V__  D__  E__
Appendix C

Student Interview

1. What do you like to do at school? (If they say "play," ask: What do you like to play with at school?)
2. If you could have anything at all, what would you want?
3. (pointing to the computer) What is this called? (pretest)
4. Do you have a computer at home? (pretest)
5. Have you used the computer at school before? (pretest)
6. Tell me how it works. (pretest)
   Tell me how the Apple II Plus computer works. (posttest)
7. What do you do when you work with it?
8. How do the pictures get on the computer's TV screen?
9. What do you think the computer is?
10. How do you think the computer thinks?
11. What do you do when you think?
12. Pretend the computer can understand what you say to it.
   What would you tell it?
13. How do you feel when you work with the computer?
14. What do you like about it? What is fun about it?
15. What do you not like about the computer? What do you think is hard?
Appendix D

Parent Questionnaire

Child's Name: ____________________________________________

1. Has your child talked to you about working with the computer?  yes  no

   Does he/she speak about their work with the computer rarely?__ occasionally?__ often?__ constantly?__

2. What are some of the comments your child has made about working with the computer, the Turtle, the floppy disks, or programming in general?

3. Have you noticed any outward changes in your child's behavior at home since she/he started working with the computer? (Is it reflected in their play, or their drawing and writing activities?) Please share any experiences you remember.

4. What kinds of feelings has your child expressed at home about working with the computer? (circle the ones that apply)

   anxiety  happiness  frustration  failure
   success  excitement  fear  competence

   Others? Please write down what experience with the computer sparked that feeling if a particular instance(s) comes to mind.

5. Overall, how do you think your child feels about working with the computer?

6. Other remarks (on back).
Appendix E
Programming Tests

Dot Tasks

1. 

2. 

3. 

4. 

(a) 

(b) 

Δ 

(c) 

(d) 

Shape Procedures

Using TO, write procedures for:

1. 

2. 

3. 
Appendix F

Programming Commands for Dot Tasks

TO FINALTEST

TO POSTTESTA
DOT [-50 0]
DOT [0 0]
DOT [0 50]
DOT [50 50]
PU
SETPOS [-50 0]
SETH 0
PD
END

TO POSTTESTB
DOT [-20 10]
DOT [10 10]
DOT [10 -20]
DOT [-20 -20]
PU
SETPOS [-20 -20]
SETH 0
PD
END

TO POSTTESTC
DOT [-30 0]
DOT [30 0]
DOT [-30 30]
DOT [30 30]
PU
SETPOS [-30 30]
SETH 0
PD
END

TO POSTTESTD
DOT [0 60]
DOT [-70 0]
DOT [35 35]
DOT [100 0]
PU
SETPOS [0 0]
SETH 0
PD
END

END
Appendix G

Correct Resolution of Dot Tasks

1. POSTTESTA
   RT 90
   FD 50
   LT 90
   FD 50
   RT 90
   FD 50

2. POSTTESTB
   FD 30
   RT 90
   FD 30
   RT 90
   FD 30
   RT 90
   FD 30

3. POSTTESTC
   RT 90
   FD 60
   RT 90
   FD 30
   RT 90
   FD 60
   RT 90
   FD 30

4. POSTTESTD
   (a) FD 60
   (b) LT 90
       FD 70
   (c) RT 90
       FD 100
   (d) RT 45
       FD 48
### Appendix H

**TABLE XII**

ANALYSIS OF PROGRAM NAMES

<table>
<thead>
<tr>
<th>Student</th>
<th>Program Name</th>
<th>Descriptive*</th>
<th>Abbreviated*</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1. STEPS</td>
<td>X</td>
<td>X (triangles)</td>
</tr>
<tr>
<td></td>
<td>2. ANGLE1</td>
<td>. . .</td>
<td>X (pinwheel)</td>
</tr>
<tr>
<td></td>
<td>3. PIN</td>
<td>X</td>
<td>X (mouse)</td>
</tr>
<tr>
<td></td>
<td>4. M</td>
<td>. . .</td>
<td>X (circle)</td>
</tr>
<tr>
<td></td>
<td>5. C</td>
<td>X (eyes)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6. DOTS</td>
<td>X (mouth)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>7. LINE</td>
<td>. . .</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8.  S</td>
<td>X (arms)</td>
<td>X (square body)</td>
</tr>
<tr>
<td></td>
<td>9. LONG</td>
<td>X</td>
<td>X (short legs)</td>
</tr>
<tr>
<td></td>
<td>10. SH</td>
<td>X</td>
<td>X (ears)</td>
</tr>
<tr>
<td></td>
<td>11. E</td>
<td>. . .</td>
<td>X (good-by)</td>
</tr>
<tr>
<td></td>
<td>12. GOOD</td>
<td>. . .</td>
<td>X (computer)</td>
</tr>
<tr>
<td></td>
<td>13. COM</td>
<td>. . .</td>
<td></td>
</tr>
<tr>
<td>AN</td>
<td>1. RECTANGLE</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>2. CONE</td>
<td>X</td>
<td>X (window)</td>
</tr>
<tr>
<td></td>
<td>3. W</td>
<td>. . .</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. SMOKY</td>
<td>X (truck)</td>
<td>. . .</td>
</tr>
<tr>
<td>B</td>
<td>1. L</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>2. I</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>3. RECl</td>
<td>X (square)</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>4. REC2</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>5. PUZ</td>
<td>X</td>
<td>X (puzzle)</td>
</tr>
<tr>
<td>D</td>
<td>1. DRAW5</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>2. EXP</td>
<td>. . .</td>
<td>X (experiment)</td>
</tr>
<tr>
<td></td>
<td>3. DRAW4</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>4. LOG04</td>
<td>. . .</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>5. DANIELLE</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>6. DRAWD</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td></td>
<td>7. DRAWA</td>
<td>X</td>
<td>. . .</td>
</tr>
<tr>
<td>Student</td>
<td>Program Name</td>
<td>Descriptive*</td>
<td>Abbreviated*</td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
<td>--------------</td>
<td>--------------</td>
</tr>
<tr>
<td>D (Cont.)</td>
<td>8. DRAWN</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>9. DRAWI</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>10. E</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>11. DRAWL</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td>J</td>
<td>1. CARROT</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>2. CARROT11</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>3. BOX</td>
<td>X</td>
<td>X (the Turtle in the box)</td>
</tr>
<tr>
<td></td>
<td>4. BOXES</td>
<td>X</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>5. BGFR</td>
<td>X</td>
<td>X (big long flower)</td>
</tr>
<tr>
<td></td>
<td>6. LOGO9</td>
<td>X (9 steps)</td>
<td>...</td>
</tr>
<tr>
<td></td>
<td>7. SG</td>
<td>X</td>
<td>X (short long)</td>
</tr>
<tr>
<td></td>
<td>8. RTCIRCLE</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

*Words in parentheses refer to the appearance of the graphics or to the complete name of the program.*
Appendix I

Students' Questions About the Computer

1. Why is it called a computer?

2. Why did it put 99 the last time I did it? (She thought she had just typed 9.)

3. Why does it take so long? (to make circles with [FD .5 RT 1])

4. (while loading program) How come it didn't know how to LOGO9?

5. When it's finished reading, the square comes? (When the drive stops loading, the cursor appears?)

6. Who made that circle? (How did the period get in the program name on the catalog in SH.LOGO?)

7. How do you spell SAVE?

8. What's this key for? (right arrow key)

9. How do you make it go that way? (back up in EDIT)

10. Where did he go? (Turtle went offscreen)

11. Is this the part that's the label? (on the floppy disk)

12. Did you buy the computer at the store and did you have it at home?

13. Which way is left?

14. How many is a hundred?

15. Do you know everything about the computer?
Appendix J

Alexandra's Programming

1. SHAPE
2. SHAPE1
3. SHAPE0
4. STEPS* 
5. ANGLE1 (triangles)
6. PIN (pinwheel)
7. BOX
8. SQ (square)
9. TURTLE (roof)
10. D (door)
11. H* (house)
12. C (circle)
13. DOTS (eyes)
14. LINE (mouth)
15. S (square body)
16. LONG (arms)
17. SH (short legs)
18. E (ears)
19. M* (mouse)
20. COM (computer)
21. GOOD (good-by)

*Indicates a program composed of two or more subprocedures.
TO SHAPE
FD 50
RT 90
FD 50
RT 90
FD 50
RT 90
FD 50
END

TO SHAPE1
FD 90
RT 90
FD 90
RT 90
FD 90
RT 90
FD 90
END
Alexandra

SHAPE0, STEPS

```
2PO "SHAPE0
TO SHAPE0
LT 90
FD 50
FD 40
LT 90
FD 40
LT 90
FD 40
END
```

```
TO STEPS
SHAPE
SHAPE1
SHAPEO
END
```
AXELANDRA

ANGLE1, PIN

?PO "ANGLE1
TO ANGLE1
RT 10
FD 90
RT 131
FD 90
RT 130
FD 90
LT 131
FD 20
END

?PO "PIN
TO PIN
REPEAT 6 [ANGLE1]
END
Alexandra

BOX, SQ, TURTLE

?PO "BOX
TO BOX
FD 60
RT 90
FD 60
RT 90
FD 60
RT 90
FD 60
RT 90
END

?PO "SQ
TO SQ
REPEAT 4 [FD 60 RT 90]
FD 60
END

?PO "TURTLE
TO TURTLE
RT 30
FD 50
RT 110
FD 50
FD 2
FD 4
END
Alexandra
D, H

```
PD "D TO D
PU
HOME
PD
FD 4
FD 6
FD 9
FD 3
RT 90
FD 7
RT 90
FD 3
FD 9
FD 6
FD 4
DOT [3 12]
END
```

```
PU "H TO H
SD
TURTLE
D
END
```
Alexandra

C, DOTS

**PO "C
TD C
FD 80
RT 90
FD 80
RT 90
FD 80
RT 90
FD 80
END

**PO "DOTS
TD DOTB
RT 90
FD 80
BK 60
FD 50
BK 10
PU
RT 90
FD 10
PD
DOT [10 60]
PU
FD 5
FD 60
BK 7
BK 10
PD
DOT [58 60]
END
Alexandra
LINE, S

2PO "LINE
TO LINE
PU
RT 90
FD 30
RT 90
PD
FD 30
END

2PO "S
TO S
PU
LT 90
FD 70
FD 21
LT 90
LT 90
FD 5
FD 60
BK 3
PD
BK 60
BK 2
RT 90
FD 50
FE
BK 5
BK 40
BK 3
PD
FD 8
FD 9
FD 5
FD 12
LT 90
FD 55
FD 3
END
Alexandra

LONG, SH

TD LONG
BK 10
BK 4
BK 6
BK 50
FD 3
BK 1
BK 1
PE
BK 1
FD 2
FD
FD 50
HT 90
FD 40
BK 40
PU
BK 50
FD 3
FD 5
FD 5
LT 90
LT 90
FD
FD 40
END

TO SH
BK 40
LT 90
FD 50
END
Alexandra

M

?PO "M
TO M
C
DUTS
LINE
S
LONG
SH
E
END
Alexandra
COM, GOOD

?PU "COM
TO COM
PR [HI ALEXANDRA]
E
END

?PU "GOOD
TO GOOD
PR [GOOD BY ALEXANDRA]
END
Appendix K

Andrea’s Programming

1. RECTANGLE
2. CONE (snowcone)
3. W
4. PART (top part of house)
5. SMOKY (truck)
Andrea

RECTANGLE, CONE

?PO "RECTANGLE TO RECTANGLE
FD 11
RT 90
FD 7
RT 90
FD 9
RT 90
FD 6
END

?PO "CONE TO CONE
LT 35
FD 32
RT 180
FD 32
LT 135
FD 32
END
Andrea

W, PART

?PD "W
TO W
FD 10
FD 21
LT 90
FD 21
FD 10
LT 90
FD 21
FD 10
LT 90
FD 10
FD 21
LT 90
FD 15
LT 90
FD 31
LT 90
FD 21
LT 90
FD 31
LT 90
FD 21
LT 90
FD 15
LT 90
FD 19
END

?PD "PART
TO PART
LT 19
LT 21
FD 18
FD 31
FD 54
LT 15
LT 121
FD 54
FD 31
END
SMOKY

TO SMOKY
PU
FD 45
FD 21
FD 12
FD 12
LT 90
PD
FD 67
LT 90
FD 78
RT 90
FD 21
FD 12
FD 12
LT 90
FD 21
FD 21
LT 90
FD 12
LT 45
REPEAT 360 [FD .5 RT 1]
PU
FD 5
RT 55
FD 5
FD 5
FD 5
FD 5
FD 5
FD 5
FD 5
PD FD 1
FD 1
FD 5
LT 12
FD 65
LT 48
REPEAT 360 [FD .5 RT 1]
PU
RT 56 FD 35 FD 10
PD
FD 21 LT 90
FD 56
FD 21 LT 90 FD 21 FD 65 RT 90 FD 2
FD 12 FD 12 FD 1 FD 1 FD 1 FD 1 FD 1
FD 11 LT 90 FD 1 FD 11
END
Appendix L

Billy's Programming

1. L
2. I
3. REC1 (square)
4. PUZ (puzzle)
5. REC2 (rectangle)
6. B
Billy

L, I

```
CPD "L
TD L
LT 90
FD 20
RT 90
FD 43
END
```

```
CPD "1
TD 1
FD 30
FD 40
END
```
Billy

RECl, PUZ

??PU "RECl
TO RECl
FD 33
LT 90
FD 33
LT 90
FD 33
LT 90
FD 33
END

??FO "PUZ
TO PUZ
REPEAT 4 [RECl]
END
Billy

```
?PU "REC2
TO REC2
FD 6
FD 24
LT 90
FD 24
FD 40
LT 90
FD 6
FD 24
LT 90
FD 24
FD 40
END
```
Appendix M

Danielle's Programming

1. E
2. DRAWA
3. DRAW4
4. DRAW5
5. EXP (experiment)
6. LOGO4
7. DRAWL
8. S (square)
9. TRI (triangle)
10. DRAWN* (house)
11. DRAWD
12. DRAWI
13. DRAWN
14. DANIELLE

*Indicates a program composed of two or more subprocedures.
Danielle

E, DRAWA

?PDU "E
TO E
LT 90
FD 40
LT 90
FD 40
LT 90
FD 40
LT 180
FD 40
RT 90
FD 20
RT 90
FD 30
END

?PDU "DRAWA
TO DRAWA
LT 35
FD 36
LT 100
FD 36
LT 180
FD 20
RT 45
FD 20
END
Danielle

DRAW4, DRAW5

2PO "DRAW4
TO DRAW4
FD 43
LT 180
FD 19
RT 90
BK 2
FD 26
RT 90
FD 20
END

2PO "DRAW5
TO DRAW5
RT 130
REPEAT 13 LFD 10 LT 203
LT 5
FD 5
RT 135
FD 40
RT 90
FD 41
END
Danielle

EXP, LOGO4

DEF EXP
TO EXP
REPEAT 4 [DRAW5]
END

DEF LOGO4
TO LOGO4
REPEAT 30 [DRAW4]
END
Danielle

DRAWL, S

PO "DRAWL
TO DRAWL
LT 90
FD 40
RT 90
FD 40
END

PO "S
TO S
BK 70
LT 90
FD 70
RT 90
FD 70
RT 90
FD 70
END
Danielle

TRI, DRAWH

:PU "TRI
TU TRI
HOME
LT 59
FD 70
PE
BK 6
BK 9
PD
LT 104
FD 57
END

:PU "DRAWH
TU DRAWH
S
TRI
END
Danielle

drawd

\begin{verbatim}
PU "DRAWD
TO DRAWD
BK 89
FD 51
BK 9
BK 20
BK 6
BK 3
BK 2
RT 90
LT 9
FD 9
REPEAT 18 [LT 9 FD 9]
LT 90
LF 3
LT 5
LT 1
FD 10
FD 10
FD 9
FD 10
END
\end{verbatim}
Danielle

DRAWI, DRAWN
Danielle

PD "DANIELLE TO DANIELLE"
PU
LT 90
FD 130
SETH 0
FD 70
PU
DRAW1
PU
HUME
LT 90
FD 20
FD 10
FD 10
SETH 0
PU
DRAWA
PU
FD 9
FD 9
SETH 0
BK 9
BK 2
BK 4
BK 1
PD
DRAWA
PU
RT 90
FD 18
SETH 0
PU
DRAW1
FD 40
RT 90
PU
FD 18
FD 30
SETH 0
PD
E
PU
FD 50
SETH 0
BK 40
FD 20
PD

DRAWL
PU
BK 40
RT 90
FD 40
FD 40
SETH 0
PD
DRAWL
PU
BK 80
BK 10
BK 10
RT 90
FD 40
SETH 0
PD
FD 40
PD
E
END
Appendix N

Jonathan's Programming

1. CARROT
2. BOX
3. BOXES
4. CARROT11
5. SPOOK
6. PART
7. SH* (spookhouse)
8. LOGO9
9. SG (short long)
10. RTCIRCLE
11. BGFR* (big long flower)

*Indicates a program composed of two or more subprocedures.
Jonathan
CARROT, BOX

2F0 "CARROT
TO CARROT
RT 90
FD 18
LT 100
FD 35
LT 150
FD 36
END

2F0 "BOX
TO BOX
FD 100
LT 90
FD 100
LT 90
FD 100
LT 90
FD 100
LT 90
FD 100
LT 90
FD 50
LT 90
FD 50
RT 180
END
Jonathan

BOXES, CARROT11

```cpp
'PO "BOXES
  TO BOXES
  REPEAT 18 [BOX]
  END

'PO "CARROT11
  TO CARROT11
  REPEAT 11 [CARROT]
  END
```
Jonathan

SPOOK, PART

2PO "SPOOK
TO SPOOK
BK 30
LT 90
FD 62
RT 90
FD 86
RT 40
FD 30
RT 60
END

2PO "PART
TO PART
PU
HUME
FD FD 30
FD 2
FD 4
FD 4
LT 6
LT 6
LT 10
LT 20
FD 30
FD 2
FD 3
FD 3
FD 3
LT 3
LT 5
LT 45
FD 3
FD 3
FD 1
FD 1
FD 2
FD 3
END
Jonathan
SH, LOG09

?PO "SH
TO SH
SPO0K
PART
END

?PO "LOG09
TO LOG09
BK 6
BK 9
BK 20
BK 20
FD 20
LT 21
LT 21
FD 12
REPEAT 4 [LT 21 FD 12]
LT 12
LT 12
LT 12
LT 12
LT 12
LT 12
LT 12
REPEAT 4 [LT 21 FD 12]
END
Jonathan

?PU "SG
TO SG
PU
FD 3
FD 3
LT 3
LT 7
LT 68
RT 5
RT 12
RT 12
FD 3
FD 3
FD 3
FD 3
FD 3
FD
FD 10
FD 45
END

?PU "RTCIRCLE
TO RTCIRCLE
LT 10
LT 10
LT 45
REPEAT 360 [FD .5 RT 1] RT 78
RT 23
PU
FD 12
RT 1
RT 1
FD
REPEAT 360 [FD .1 RT 1]
REPEAT 360 [BK .1 RT 1]
HT
END
Jonathan

BGFR
BIBLIOGRAPHY

Books


Poirot, J. L., & Adams, R. C. Forty easy steps to programming in BASIC & LOGO. Austin: Sterling Swift, 1983.


Articles


Marbach et al., W. D. To each his own computer. *Newsweek*, 1982, 99(8), 50-56.


Reports


Unpublished Materials


Newspapers