INFORMATION TO USERS

This reproduction was made from a copy of a document sent to us for microfilming. While the most advanced technology has been used to photograph and reproduce this document, the quality of the reproduction is heavily dependent upon the quality of the material submitted.

The following explanation of techniques is provided to help clarify markings or notations which may appear on this reproduction.

1. The sign or “target” for pages apparently lacking from the document photographed is “Missing Page(s)”. If it was possible to obtain the missing page(s) or section, they are spliced into the film along with adjacent pages. This may have necessitated cutting through an image and duplicating adjacent pages to assure complete continuity.

2. When an image on the film is obliterated with a round black mark, it is an indication of either blurred copy because of movement during exposure, duplicate copy, or copyrighted materials that should not have been filmed. For blurred pages, a good image of the page can be found in the adjacent frame. If copyrighted materials were deleted, a target note will appear listing the pages in the adjacent frame.

3. When a map, drawing or chart, etc., is part of the material being photographed, a definite method of “sectioning” the material has been followed. It is customary to begin filming at the upper left hand corner of a large sheet and to continue from left to right in equal sections with small overlaps. If necessary, sectioning is continued again—beginning below the first row and continuing on until complete.

4. For illustrations that cannot be satisfactorily reproduced by xerographic means, photographic prints can be purchased at additional cost and inserted into your xerographic copy. These prints are available upon request from the Dissertations Customer Services Department.

5. Some pages in any document may have indistinct print. In all cases the best available copy has been filmed.
Oprea, Janeal Mika

THE EFFECTS OF COMPUTER PROGRAMMING ON A STUDENT'S MATHEMATICAL GENERALIZATION AND UNDERSTANDING OF VARIABLES

The Ohio State University

University Microfilms International

300 N. Zeeb Road, Ann Arbor, MI 48106

PH.D. 1984
PLEASE NOTE:

In all cases this material has been filmed in the best possible way from the available copy.
Problems encountered with this document have been identified here with a check mark. 

1. Glossy photographs or pages 
2. Colored illustrations, paper or print 
3. Photographs with dark background 
4. Illustrations are poor copy 
5. Pages with black marks, not original copy 
6. Print shows through as there is text on both sides of page 
7. Indistinct, broken or small print on several pages 
8. Print exceeds margin requirements 
9. Tightly bound copy with print lost in spine 
10. Computer printout pages with indistinct print 
11. Page(s) lacking when material received, and not available from school or author.
12. Page(s) seem to be missing in numbering only as text follows.
13. Two pages numbered . Text follows.
14. Curling and wrinkled pages 
15. Other

University Microfilms International
THE EFFECTS OF COMPUTER PROGRAMMING ON A STUDENT'S
MATHEMATICAL GENERALIZATION AND UNDERSTANDING OF VARIABLES

DISSERTATION

Presented in Partial Fulfillment of the Requirements for
the Degree doctor of Philosophy in the Graduate
School of The Ohio State University

by

Janeal Mika Oprea, B.A., M.S.

*****

The Ohio State University

1984

Reading Committee:
Richard J. Shumway
Suzanne K. Damarin
Bert K. Waits
Grayson H. Wheatley
Arthur L. White

Approved By

Richard J. Shumway
Adviser

Department of Theory and Practice
(Mathematics Education)
ACKNOWLEDGEMENTS

For their love and faith, this work is dedicated to my parents, Jim and Joanna Mika.

I wish to thank my adviser, Richard Shumway. In his own unique way, Dick has helped me develop the skills and perspective needed for educational research.

When I began working at Purdue, I asked Grayson Wheatley to be my "adviser in residence"; for his acceptance I am grateful. He willingly found time in his busy schedule for countless hours of discussion as well as reading and critiquing my numerous revisions.

Throughout my graduate studies, Suzanne Damarin provided me with guidance and educational opportunities. Through Suzanne, I became involved with the TABS-math project which furnished me with invaluable first hand knowledge of the use of microcomputers in education.

I am completely and forever indebted to my husband, John, for without his support and tolerance this project may have never been completed.
VITA

May 24, 1955.................. Born – Garfield Heights, Ohio

1977............................. B.A., Mathematics, Ohio Northern
                                 University, Ada, Ohio

1977-1978..................... High School Mathematics/General Science
                                 Teacher, Wheelersburg, Ohio

1978-1982..................... Teaching Associate, Mathematics and
                                 Mathematics Education Department, The
                                 Ohio State University, Columbus, Ohio

1980............................. M.S., Mathematics, The Ohio State
                                 University, Columbus, Ohio

1981-1983..................... Research Associate, Technology and Basic
                                 Skills in Mathematics Project, The Ohio
                                 State University, Columbus, Ohio

1983-present................... Instructor, Mathematics and Mathematics
                                 Education, Purdue University, West
                                 Lafayette, Indiana

PUBLICATIONS

Wilson, P.S. and Oprea, J. M. The computer as an aid in mathematical
concept building. Paper presented at the annual meeting of the
Research Council of Diagnostic and Prescriptive Mathematics,
Bowling Green, Ohio, April 1983 (To appear in the 1984
Proceedings).
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACKNOWLEDGEMENTS</td>
<td>ii</td>
</tr>
<tr>
<td>VITA</td>
<td>iii</td>
</tr>
<tr>
<td>LIST OF TABLES</td>
<td>vi</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>ix</td>
</tr>
<tr>
<td><strong>CHAPTERS</strong></td>
<td></td>
</tr>
<tr>
<td>I. INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>Need</td>
<td>3</td>
</tr>
<tr>
<td>Problem Statement</td>
<td>5</td>
</tr>
<tr>
<td>Assumptions</td>
<td>6</td>
</tr>
<tr>
<td>Limitations</td>
<td>6</td>
</tr>
<tr>
<td>Definition of Terms</td>
<td>7</td>
</tr>
<tr>
<td>II. LITERATURE REVIEW</td>
<td>10</td>
</tr>
<tr>
<td>Computer Programming</td>
<td>10</td>
</tr>
<tr>
<td>Problem Solving</td>
<td>22</td>
</tr>
<tr>
<td>Generalization</td>
<td>28</td>
</tr>
<tr>
<td>III. METHODS AND PROCEDURES</td>
<td>32</td>
</tr>
<tr>
<td>Overview</td>
<td>23</td>
</tr>
<tr>
<td>Pilot Study</td>
<td>34</td>
</tr>
<tr>
<td>Treatment</td>
<td>35</td>
</tr>
<tr>
<td>Subjects</td>
<td>42</td>
</tr>
<tr>
<td>Procedures</td>
<td>45</td>
</tr>
<tr>
<td>Instruments</td>
<td>47</td>
</tr>
<tr>
<td>Statistical Analysis</td>
<td>58</td>
</tr>
</tbody>
</table>
IV. RESULTS
   Empirical .......................................................... 64
   Clinical ............................................................ 84

V. CONCLUSIONS AND DISCUSSION .................................. 121

REFERENCES ............................................................ 133

APPENDICES
   A. Instruments ...................................................... 140
   B. Pilot Study ...................................................... 152
   C. Programming Activities - Treatment W ...................... 174
   D. Programming Activities - Treatment E ...................... 197
   E. Interview Script and Materials .............................. 216
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Differences of Programming Instruction</td>
<td>42</td>
</tr>
<tr>
<td>2. Summary of Treatment Groups by Sex and Class</td>
<td>43</td>
</tr>
<tr>
<td>3. Summary of Control Groups by Sex and Class</td>
<td>44</td>
</tr>
<tr>
<td>4. Correspondence of Doll's Generalization Levels to Bloom's Taxonomy</td>
<td>48</td>
</tr>
<tr>
<td>5. Generalization Test Items by Behavioral Objectives and Mathematical Content</td>
<td>51</td>
</tr>
<tr>
<td>6. Variables Test Items by Behavioral Objectives and Complexity Levels</td>
<td>55</td>
</tr>
<tr>
<td>7. Pretesting Procedures for the Control Groups</td>
<td>57</td>
</tr>
<tr>
<td>8. Summary of Treatment and Testing</td>
<td>58</td>
</tr>
<tr>
<td>9. Modified Experimental Design</td>
<td>59</td>
</tr>
<tr>
<td>10. Reliabilities of the Generalization, Programming, and Variable Instruments</td>
<td>66</td>
</tr>
<tr>
<td>11. Eigenvalues and Percentages of Variance Explained By the Factors of the Rotated Factor Matrix</td>
<td>68</td>
</tr>
<tr>
<td>12. Varimax Rotated Factor Matrix</td>
<td>69</td>
</tr>
<tr>
<td>13. Reliabilities of the Valid Subsets of the Generalization, Programming, and Variables Instruments</td>
<td>71</td>
</tr>
<tr>
<td>14. Summary of the Background Data by Group</td>
<td>72</td>
</tr>
<tr>
<td>15. Means and Standard Deviations of Pretest Scores by Group</td>
<td>73</td>
</tr>
<tr>
<td>16. ANOVA Results of Mean Pretest Scores by Treatment Group</td>
<td>73</td>
</tr>
<tr>
<td>Table</td>
<td>Page</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>17. Hartley's Test of Homogeneity of Variance on Mean Pretest Scores</td>
<td>74</td>
</tr>
<tr>
<td>18. Means and Standard Deviations of the Control Groups Pre- and Posttest Scores</td>
<td>75</td>
</tr>
<tr>
<td>19. ANOVA Results of Mean Posttest Scores by Control Group</td>
<td>76</td>
</tr>
<tr>
<td>20. Hartley's Test of Homogeneity of Variance on the Control Groups' Mean Posttest Scores</td>
<td>76</td>
</tr>
<tr>
<td>21. Means and Standard Deviations of Posttest Scores by Group</td>
<td>77</td>
</tr>
<tr>
<td>22. Univariate F-Tests of Individual Mean Posttest Scores by Group</td>
<td>77</td>
</tr>
<tr>
<td>23. Stepwise Multiple Regression of Posttest Scores</td>
<td>78</td>
</tr>
<tr>
<td>24. Stepwise Multiple Regression of Posttest Scores of Treatment Groups W and E</td>
<td>79</td>
</tr>
<tr>
<td>25. Pearson Correlation Coefficients of Background Variables</td>
<td>81</td>
</tr>
<tr>
<td>26. Adjusted Means of Posttest Scores by Group</td>
<td>82</td>
</tr>
<tr>
<td>27. Univariate F-tests of Individual Adjusted Mean Posttest Scores</td>
<td>82</td>
</tr>
<tr>
<td>28. Results of Dunn's Test</td>
<td>83</td>
</tr>
<tr>
<td>29. Profiles of Case Study Subjects</td>
<td>87</td>
</tr>
<tr>
<td>30. Summary of Field Test Groups by Grade and Sex</td>
<td>153</td>
</tr>
<tr>
<td>31. Mean Ages and Programming Pretest Scores of the Computer Camp Subjects by Session and Subsection</td>
<td>156</td>
</tr>
<tr>
<td>32. Descriptive Statistics of the Various Versions of the Generalization Instrument</td>
<td>170</td>
</tr>
<tr>
<td>33. Descriptive Statistics of the Various Versions of the Programming Instrument</td>
<td>177</td>
</tr>
</tbody>
</table>
34. Descriptive Statistics of the Various Versions of the Variable Instrument.............178

35. Profile of Computer Camp Interview Participants..............181
LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Program for computing the LCM of two numbers</td>
<td>13</td>
</tr>
<tr>
<td>2. Skemp's description of mathematical generalization</td>
<td>29</td>
</tr>
<tr>
<td>3. Model of the relationship of real world problems and mathematical systems</td>
<td>35</td>
</tr>
<tr>
<td>4. Sample introductory program for Treatment W</td>
<td>37</td>
</tr>
<tr>
<td>5. Sample follow-up program for Treatment W</td>
<td>37</td>
</tr>
<tr>
<td>6. Sample introductory program for Treatment E</td>
<td>40</td>
</tr>
<tr>
<td>7. Sample follow-up activity for Treatment E</td>
<td>41</td>
</tr>
<tr>
<td>8. Seating and computer arrangement of treatment classroom</td>
<td>46</td>
</tr>
<tr>
<td>9. Subtest 1 of the Generalization instrument</td>
<td>49</td>
</tr>
<tr>
<td>10. Examples of one-step and two-step problems from the Variable instrument</td>
<td>54</td>
</tr>
<tr>
<td>11. A two-step problems which does not involve arithmetic computation</td>
<td>55</td>
</tr>
<tr>
<td>12. Matt’s pre- and posttest answers for problem 7 of the Variable instrument</td>
<td>88</td>
</tr>
<tr>
<td>13. Sample program from the computer camp</td>
<td>157</td>
</tr>
<tr>
<td>14. Sample items from the second version of the Generalization instrument</td>
<td>161</td>
</tr>
<tr>
<td>15. Sample items from the third version of the Generalization instrument</td>
<td>162</td>
</tr>
</tbody>
</table>
16. Sample items from the fourth version of the Generalization instrument..........................163

17. Sample items from the first version of the Programming instrument.............................166

18. Sample items from the second version of the Variable instrument.................................169

19. Interview questions used at the computer camp.................................181
CHAPTER ONE
INTRODUCTION

The intent was to investigate the effects of computer programming instruction on students' understanding of variables and on their mathematical generalization, a cognitive process associated with mathematical problem solving ability. In addition, the effects of different instructional methodologies on students' mathematical generalization and understanding of variables were studied.

Since the advent of computers, mathematics educators have often noted the apparent connections between the thinking involved in computer programming and various aspects of mathematical cognition (e.g. Hatfield, 1979; Feurzig and Papert, 1968). The processes of analyzing, simplifying, generalizing, and conjecturing are among the possible aspects of cognition which might be enhanced through effective programming. Mathematical generalization is the ability to detect structure and similarity in externally dissimilar situations (Krutetskii, 1976). While generalization is only one of the mental processes that correlate positively with mathematical aptitude, many mathematicians feel it is an extremely important attribute of mathematical maturity (Shumway, 1968; Krutetskii, 1976).
Presently, in the elementary grades, instruction in computer programming is part of the mathematics curriculum. However, the objective of this instruction is the development of programming skills and not the enhancement of mathematics learning. The initial focus is on the elementary commands of the programming language rather than the structural commands. Simple programs are often used to illustrate the function of the commands. These programs are often devoid of any meaningful mathematical content. While this elemental approach may foster competent programming, it may not necessarily enhance mathematical thought. An alternate approach, the wholistic method, begins instruction at the whole program level with students learning the fundamentals of the computer language through the programs themselves. Based on this approach, the researcher designed classroom activities which posed mathematically relevant problems and covered programming topics as needed for completion of each assignment. It was hypothesized the Wholistic treatment group (Group W) would have a greater increase in mathematical generalization than the Elemental treatment group (Group E).

Sixth grade classes of two elementary schools were the treatment groups. The classes of one school acted as a control while the students of the other were randomly assigned to Group W or Group E. For 6 weeks, Groups W and E studied computer programming during 60-90 minutes sessions 2 or 3 times a week. Overall, these students participated in approximately 20 hours of programming instruction. Pre- and posttests assessing programming ability (PROG), mathematical
generalization (GEN), and understanding of variables (VAR) were administered. In addition, five students from each treatment group were selected for 40 minute interviews to evaluate the treatment effect.

NEED

The potential positive effects computer programming could have on mathematics learning are numerous and varied, ranging from the enrichment of mathematical concept learning to the enhancement of deductive and inductive reasoning. Perhaps the boldest assertion is made by Papert (1980) who suggests the computer is the medium for learning how to learn. Most of these claims, however, are based on rational argument, individual observation, and the experiences of "expert witnesses" such as practitioners and educators - and not supported by systematic empirical research (De Corte, 1984; Pea and Kurland, 1983).

Mathematics educators have, in particular, often noted the apparent connections between the thinking involved in building, testing, correcting, and refining a computer program and various aspects of mathematical cognition, such as generalization (e.g. Hatfield, 1979; Feurzig and Papert, 1968). After spending a year studying the influence of computer experience on children's cognitive development, instructional psychologist De Corte (1984) summarized his findings as follows:
Does learning to program result in the acquisition of conceptual knowledge and thinking methods which can be transferred to other content domains? I have come to the conclusion that, at present, there is only scarce evidence supporting the transfer hypothesis. However, I must at the same time admit that there is also no counterevidence, and that there are few good empirical studies relating to this hypothesis. (p. 1)

While generalization is only one of the mental processes that correlate positively with mathematical aptitude, many mathematicians feel it is an extremely important attribute of mathematical maturity (Shumway, 1968; Krutetskii, 1976). Krutetskii states, "an ability to generalize mathematical material, to detect what is of chief importance, to [ignore] the irrelevant, and to see what is common in what is externally different" is a basic characteristic of mathematical thought (p. 87). In fact, Krutetskii believes "abstractions and generalizations constitute the essence of mathematics and mathematical thinking" (p. 86). With this in mind, the microcomputer could have a significant impact on the mathematics curriculum if there were convincing evidence that learning to program resulted in the enhancement of a student's mathematical generalization. However, to date, only one study has addressed this specific issue and the results were inconclusive (Foster, 1972).

Mathematics educators have also asserted a student's understanding of fundamental mathematical concepts, such as variable and equation, is related to programming ability. Yet, again there has not been a great deal of research focusing on the relationship between programming and the understanding of variables. One such study involved fifth graders learning Logo (Milner, 1973) while the other involved college students...
learning BASIC (Clement, 1982). Both studies found a correspondence between programming ability and understanding of variables. The result that sixth grade students' understanding of variables improved after programming instruction, would reinforce the findings of Milner and Clement and generalize the theory to a larger population.

The educators cited above suggest mathematical generalization and understanding of variables occupy important places in the mathematical education process. Furthermore, it is clear, educators as well as the public want to believe the microcomputer is an effective instructional aid. However, there is a disturbing lack of evidence that learning to program results in the acquisition of generalizable and transferable conceptual knowledge and thinking processes. With this in mind, well-designed, empirical research which describes the effects of computer programming on mathematical generalization not only fills a gap in the literature, but might well have a significant impact on the mathematics curriculum.

PROBLEM STATEMENT

The purpose was to investigate the relationship of computer programming to students' mathematical generalization and understanding of variables. Furthermore, the effects of two different instructional methodologies on students' mathematical generalization and understanding of variables were studied.
ASSUMPTIONS

1. The three treatment groups had the same distribution with respect to general intelligence and programming experience prior to the treatment.

2. Most subjects were familiar with the mathematical terminology and concepts used in the Generalization instrument and programming activities.

3. The Logo instruction the subjects received prior to the treatment had a limited but uniform effect on the subjects' mathematical generalization and computer programming ability.

LIMITATIONS

1. Generalization of the population is limited to sixth grade students from a small midwestern town. Since the exact effects of the prior Logo instruction are not known, the population is further limited to students with experience in this language.

2. The results are specific to the programming language BASIC.

3. While the treatments were administered to groups, the unit of analysis is the single student.
DEFINITION OF TERMS

Mathematical Generalization - a cognitive process. The process of extending a universe to a class containing an isomorphic image of the original restricted class. For example, the extension of the natural numbers to the integers or the rational numbers to rational expressions. Abstraction, on the other hand, is the process of drawing from a number of different situations something which is common to them all. During the act of problem solving, these two processes often overlap. For the Generalization instrument, levels of generalization were approximated by the following behaviors:

1. See a pattern
2. Utilize a pattern
3. Extend a pattern
4. Make a generalization

A further discussion of mathematical generalization can be found in Chapter Two.

Computer Programming Instruction -

A. Wholistic Approach - Begin instruction at the whole program level. The focus is on mathematically relevant problems and commands are introduced only as needed to solve the problem.
B. Elemental Approach - The focus is on the individual BASIC commands and how they function. Begin with the simplest commands and add to these until the student is capable of programming complex problems.

Understanding of Variables - The instrument to measure understanding of variables assessed achievement of the following behavioral objectives:

1. Evaluate a simple algebraic expression
2. Translate verbal statements to symbolic equations
3. Recognize all possible equations for a verbal statement

Mathematical Ability - measured by the student's national percentile score on the mathematical application subtest of the CTBS exams. Students' mathematical ability was rated high if their CTBS score was greater than 50, otherwise they were rated as low ability.

Used Computer - A yes/no response depending on whether the student indicated on the background questionnaire that they had ever used a computer besides the three weeks of Logo instruction at school. The rating was based on the student's responses to the following background questions (see Appendix A):

1. Have you ever used a computer?
2. Check ALL the ways you have used a computer.
6. Check ALL of the following [computer] activities in which you have participated.
**Own Computer** - A yes/no response depending on whether students indicated on the background questionnaire that they had a microcomputer at home. The microcomputer must be programmable and not simply a computerized game machine (e.g. Coleco Intellivision).

**Months** - The approximate number of months the student owned a microcomputer.
CHAPTER TWO
LITERATURE REVIEW

COMPUTER PROGRAMMING

The great advances in computer technology this past decade are widely acclaimed. The development of the microcomputer offers schools an excellent opportunity to expand the use of computers in the classroom (Hansen and Zweng, 1984; Fey, 1984; Braun, 1981; Damarin, 1981). Mathematics educators have asserted the computer is a powerful educational tool and are encouraging schools to use computers. In An Agenda for Action, the National Council of Teachers of Mathematics recommended that "mathematics programs take full advantage of the power of calculators and computers at all grade levels" (1980). Similarly, in its report to the NSB Commission on Precollege Education in Mathematics, Science and Technology, the Conference Board on Mathematical Sciences (1982) states:

With regard to elementary and middle school mathematics, in summary, we recommend: that calculators and computers be introduced in the mathematics classroom at the earliest grade practical. Calculators and computers should be utilized to enhance the understanding of [mathematics] as well as the learning of problem solving (p.1).

The "proper use" of computers in education is a subject of considerable debate in the current literature (Hansen and Zweng, 1984; Braun, 1981; Bitter, 1982; Damarin, 1981). Zinn (1979) distinguished four instructional uses of computers:
1. COMPUTER AWARENESS—learning about computers; capabilities and limitations of computers, how computers effect our lives.

2. COMPUTER ASSISTED INSTRUCTION—learning through computers; tutorials, drill and practice, simulations, educational games.

3. COMPUTER PROGRAMMING—learning with the computer.

4. COMPUTER MANAGED INSTRUCTION—learning with the support of the computer.

Computer programming and its relationship to mathematical generalization, a cognitive process associated with mathematical ability, was one of the focuses of this investigation. While research on the effectiveness of most of the educational uses of the computer is sparse, there is limited evidence that programming aids mathematics learning (Blume, 1984; Hatfield and Kieren, 1972; Johnson, 1966).

The potential positive effects computer programming could have on mathematics learning are numerous and varied, ranging from the enrichment of mathematical concept learning to the enhancement of deductive and inductive reasoning. According to Fey (1984),

There are persuasive arguments that the act of writing programs will help students attain a deeper understanding of mathematical ideas and that the thinking habits learned in good programming practice will be powerful problem solving heuristics (p. 2).

Perhaps the boldest assertion is made by Papert (1980) who suggests that the computer is the medium for learning how to learn. However, most of these claims are based on rational argument, individual observation, and the experience of "expert witnesses"—such as practitioners and educators—and not supported by systematic empirical research.
To effectively research the relationship between computer programming experiences and mathematical ability, the focus of the inquiry must be narrowed. It is as Camp and Marchionini (1984) stated, there is a role for programming in mathematics education. But as you consider the options, make clear distinction between the study of programming, which belongs to the domain of computer literacy and computer science, and the use of programming to achieve learning objectives in mathematics (p.118).

Shumway (1984) classified the various potential effects of programming by the similarity of the activity of programming to the specific mathematical activity:

1. Computer Literacy
2. Specific Mathematical Concepts
3. General Mathematical Concepts
4. Mathematical Thinking
5. Problem Solving
6. Logical Reasoning

For Shumway, the term computer literacy encompasses learning specific programming commands as well as the capabilities and limitations of computers. Specific mathematical concepts are those that can be the topic of a program (e.g. LCM, integration, exponential growth, and counting by fives). On the other hand, general mathematical concepts are fundamental concepts such as variable, recursion, and sequence that are invariant across programs. The term mathematical thinking refers to the higher level cognitive processes that are positively correlated to mathematical achievement. Generalization and conjecturing are two such abilities.
This classification is based on the degree of transfer required by the student from the programming activity to the mathematical activity. For example, programming may be used to augment instruction of certain specific mathematical concepts, such as least common multiple (LCM). It is felt that by constructing a successful program for computing the LCM, the student must complete a careful analysis of the concept definition (see Figure 1). In this case, the activity of programming is simply an alternate strategy for causing the student to learn the concept.

```
10 INPUT A
20 INPUT B
30 FOR L = A TO A+B STEP A
40 IF INT(L/B) = L/B THEN 60
50 NEXT L
60 PRINT "LCM OF ",A," ",B," = ",L
70 END
```

Figure 1. Program for computing the LCM of two numbers

On the other hand, some general mathematical concepts, such as variable and equation, are fundamental in both mathematics and programming but are not necessarily taught in a single programming activity. The relationship between programming and mathematical ability is not as direct. There are certain cognitive processes that are related to mathematical ability - such as generalization, conjecturing, and analyzing - which seem to be enhanced through effective student programming but are never directly taught. This classification shall be used to discuss past studies of the effects of programming on
mathematics achievement.

1. **Programming and Computer Literacy**

The National Center for Education Statistics has recently released figures revealing that the use of microcomputers in schools tripled from Fall 1980 to Spring 1983. There are now 120,000 microcomputers for students in 35% of the country's public schools: 22% of these are in elementary schools and 64% are located in secondary schools (Reed, 1983). Results of the National Assessment of Educational Progress reveals that while many of these computers are located in the mathematics classrooms, they are primarily used for computer literacy (Carpenter, 1983). From the considerable number of students that are being taught computer programming, one might conclude that programming aptitude and skills can be developed in any student at any grade level. There is little mathematics education research to support this assumption. Kieren (1968) found that although all the eleventh grade mathematics students in his two year study achieved passable programming skills, there was a wide range in aptitude, skill, and interest in programming among the subjects. King (1972) and Hatfield (1969) also found programming aptitude and skill varied among their junior high subjects.

The interactive capability and the graphics mode of the microcomputer may have made programming a more concrete activity. Some mathematics educators believe these improvements make it possible for younger children to learn to program computers (Shumway, 1983; Damarin,
Papert (1980) claims very young children can learn to program using Logo but his claims are based on informal clinical studies with a few students.

When the focus of computer instruction in a mathematics curriculum is on the teaching of the programming language and programming skills, Wilkinson (1984) believes "the computer has been treated as the object of instruction rather than as a tool for instruction" (p. 404). He asserts that in such instances computer usage may have a weakening, not strengthening effect on the mathematics curriculum. This view that computer literacy should not be a primary focus of mathematics education research is best summarized by Hatfield (1984):

We know that students can learn to write computer programs; we must now consider the more critical questions about why our mathematics students should become engaged in such tasks (p. 2). Educators have...failed to analyze carefully the pedagogical and psychological bases for including programming tasks: the focus must be on simulating and guiding students as they construct the ideas and processes we call mathematical knowledge. Curriculum developers have included computer-programming activities as optional enrichment rather than as significant situations embodying mathematical thinking essential to the goals of the curriculum (p. 6).

2. Programming and Specific Mathematical Concepts

In the 1970's, most computer programming related mathematics education research investigated the effectiveness of teaching specific mathematical concepts by having students, grades 7-16, write programs embodying those concepts (e.g. Kieren, 1973; Hatfield, 1971; Washburn, 1969). Camp and Marchionini (1984) trace the theoretical link between computer programming and mathematical concept development to the
Research on tutoring (the basis for cross-age tutoring programs), [the fact that] teachers learn from teaching (ask any teacher), and those who practice learn by practicing. The statement is well known that programming is teaching—teaching a computer—and hence programmers may learn from programming (p. 118).

Reviews of these studies concluded that selected mathematics content, such as certain number theory concepts, is better learned when students write and run programs either embedding the concept being studied or displaying the concept in the output (Blume, 1984; Kieren, 1973). The influence of mathematical ability on these results is not clear. Contrary to the results of the seventh grade subjects, Hatfield and Kieren (1972), noted that for the eleventh grade level, average students seemed to benefit more from the computer use than the above average students. However, the results of Washburn's (1969) study of seventh, eight, and twelve grade students indicated that above average students benefitted relatively more than others.

3. Programming and General Mathematical Concepts

While most mathematics education researchers assert a student's understanding of fundamental mathematical concepts, such as variable and equation, is related to programming ability, there has been little research. Hart (1982) presented informal data that computer programming instruction may positively influence the development of variable. Milner (1973) studied the effect of instruction in Logo on fifth graders understanding of variables and showed significant results in favor of the experimental group. In a study that was an extension
of his work on college student's translating difficulties of algebra word problems, Clement (1982) showed the positive effects of programming in BASIC on these students' understanding of variables and equations.

4. Programming and Mathematical Thinking

As in the case of certain fundamental mathematical concepts, mathematics educators have often noted the apparent connections between the cognitive processes involved in computer programming and various aspects of mathematical thinking (e.g. Hatfield, 1984; Feurzig and Papert, 1968). The processes of analyzing, simplifying, particularizing, generalizing, justifying, conjecturing, and structuring are among the possible aspects of cognition which might be enhanced through effective student programming. Hatfield (1971) noted that many students would persist through several versions in refining and extending an already successful program and this would "foster inductive strategies aimed at generalizing and discovering" (p. 5). According to DeCorte (1984), these hypotheses may not be easily verified:

The history of the psychology of learning but also more recent studies show that achieving transfer of cognitive skills is not an easy matter...and transfer certainly has to be pursued explicitly. Moreover, research in cognitive psychology has recently produced robust evidence that problem-solving ability within a given domain of content strongly depends on domain-specific knowledge (p. 1).
Summarizing the research in this area, DeCorte states:

At present, there is only scarce evidence supporting the transfer hypothesis. However, I must at the same time admit that there is also no counterevidence, and that there are very few good empirical studies relating to this hypothesis (p. 1).

Foster (1972) conducted a study specifically focusing on some of these cognitive processes. Using eighth grade mathematics students, he studied the effects of programming and/or flowcharting on nine selected behaviors associated with problem solving. Two of these behaviors, proposing a hypothesis and identifying a pattern, are relevant since both involve the process of generalization. While the group learning to program did significantly better on the posttest than the control group, the difference was only marginally significant on the subtest measuring the ability to propose a hypothesis. Furthermore, there was no evidence of a difference between these two group's scores on the subtest measuring the ability to identify a pattern. The lack of significant results may be because all of the subjects were involved in twelve weeks of problem solving instruction which may have improved all students' ability to generalize.

Milojkovic (1983) studied the cognitive and motivational consequences of fifth grade students learning to program. Four treatment groups were involved in the ten week study - one group was taught Logo, a second BASIC, the third had exposure to computers through courseware, and the fourth group acted as a control. Although some cognitive subscales did show limited evidence in favor of Logo
over BASIC, the overall pattern of results for the dependent measures involving cognitive variables failed to support the major hypotheses. In general, comparable motivational advantages were evident in each of the three computer groups.

Psychological studies of expert and novice programmers seem to indicate a relationship between programming ability and mathematical generalization. These studies have revealed differences in programming-specific problem solving strategies, such as debugging. In general, the differences are based in the subjects' ability to generalize. For example, Jefferies (1982), in a comparison of the debugging strategies of novice Pascal programmers and graduate computer science students, found that "experts saw whole blocks of code as 'instantiations' of well-known problems" such as "calculating change" (p. 223). Concurring with Jefferies, Pea and Kurland (1984) concluded "Expert programmers not only have more information about the computer program domain, but remember larger, meaningful chunks of information that enable them to perceive programs and remember them more effectively than novice programmers" (p. 10).

5. Programming and Problem Solving

The interaction of computers with the learning of problem solving behaviors is a broad area of study encompassing many areas of research. Studies in the field of artificial intelligence (e.g. Goldin and Luger, 1974; Newell and Simon, 1975) have had a profound effect on the way educational researchers and psychologists view problem solving
behavior. These and related studies have led to the conceptualization of problem solving as search behavior and to the use of such concepts as state-space in problem solving research. While artificial intelligence research makes use of computers and sophisticated programming languages, it is not directly related to research of the relationship of programming and mathematics learning. An indirect contribution was the development of some of the "natural" languages, such as Logo, and SMALLTALK, that are presently being advocated by educators. For a discussion of artificial intelligence, the reader is referred to Barr and Feigenbaum (1981, 1982).

Many mathematics educators believe there is a strong relation between problem solving and computer programming (e.g. Hersberger, 1983; Davis, 1982; Piele, 1979; Feurzig and Papert, 1968). Piele (1979) states "The process of learning to program a computer is a creative and inventive activity which exercises all aspects of the problem solving process" (p.3). Similarly, Feurzig and Papert (1968) assert

The activity of programming...fosters an experimental approach toward problem solving. The use of programming language provides students with a natural framework, standard vocabulary, and a set of personal experiences for discussing mathematics (p.1).

Reviews of the literature concluded that there is initial evidence that the same heuristics are used in computer programming and in problem solving (Hersberger, 1983; Wells, 1981; Johnson and Harding, 1979). For example, Johnson and Harding (1979) found Cambridge University students who completed a computer science elective did significantly
better on the annual Tripos exams than the students who did not study computer programming. These results are relevant since the students had no prior programming experience and the Tripos exams are generally accepted as realistic and consistent assessments of mathematical problem solving ability. Further discussion of this topic can be found in the section reviewing mathematical problem solving literature.

6. Programming and Logical Reasoning

All programming languages are based on the fundamental concepts of logic. Thus many educators believe experiences in computer programming can teach the logic principles inherent in the languages (e.g. Shumway, 1984; Becker, 1982). According to Becker (1982), "programming, although requiring capacities of logical thought and abstract reasoning itself, generates intellectual growth in these skills" (p.7). A recent study of the effects of learning to program on the conditional reasoning of children did not support this claim (Seidman, 1980). The four principles of conditional logic Seidman examined were forward conditional, contraposition, inversion, and conversion. Only one of the major null hypotheses dealing with the logical principles was rejected: the experimental group did significantly better than the control group on inversion under the biconditional interpretation of the conditional statement.
PROBLEM SOLVING

During the 1970’s, researchers in mathematics education devoted more research time to problem solving than to any other topic (Suydam, 1976). Among researchers there was, however, a lack of agreement of what constitutes problem solving, how performance should be measured, and what problem solving tasks are appropriate for investigation. Kilpatrick (1978) categorized research variables according to factors associated with problem solving:

1. TASK VARIABLES - the nature of the problem.
   a) Context
   b) Structure
   c) Format

2. SUBJECT VARIABLES - characteristics of the individual problem solver.
   a) Organismic - not open to change or experimental manipulation, e.g. sex, age, SES.
   b) Trait - modifiable, e.g. cognitive style, attitude, generalization ability.
   c) Instructional history

3. SITUATION VARIABLES - environmental features external to the problem and problem solver.
   a) Physical setting
   b) Psychological setting

According to Kilpatrick, any problem solving event involves a complex interaction among the variables describing these three categories. No attempt has been made here to provide an exhaustive review of the problem solving literature. Rather Kilpatrick’s categories will be used to discuss those problem solving studies relevant to mathematics
education investigations of generalization and/or computer programming.

1. **Task Variables**

A primary reason it is difficult to draw any general conclusions about mathematical problem solving is that a wide diversity of problems have been used in research (Lester, 1980). Tasks range from direct computations to puzzle problems, like the Tower of Hanoi, to long-term group projects. Lester (1980) stated there is a general agreement that the task confronting a problem solver has a significant influence on performance. Consequently, a substantial amount of research has focussed on various characteristics of problem tasks. Unfortunately, no set of variables has been clearly established as the most important determinants of problem difficulty. The following are a few of the conclusions Lester was able to draw from the problem task research.

1. Beyond the most obvious statement that children cannot solve a problem if they cannot read it, there is conflicting evidence regarding the effect of reading ability on problem solving performance.
2. Evidence suggests that linguistics predictor variables (e.g. problem length, punctuation) are age or grade level specific.
3. The importance of computational variables as predictors decreases from being very important in the intermediate grades to having little value as predictors for college students (p. 291).

As shall be seen in the next section, there is evidence of the positive relationship between mathematical generalization and problem solving ability. Therefore, the results of task variable research can serve as guidelines for the development of an instrument that measures
2. **Subject Variable**

Although only a small portion of problem solving research has been devoted exclusively or even primarily to a consideration of subject variables, most studies do consider various student characteristics as independent variables. In a recent analysis of variables and methodologies in problem solving research, Kilpatrick (1978) classified subject variables as being either organismic or trait variables. Organismic variables, such as sex, age, and socioeconomic status, are generally used to describe subjects. Trait variables, such as abilities, attitudes, and personality factors, appear open for modification.

While investigating the relationship between problem solving ability and the perception of problem structure, Krutetskii (1976) found that good problem solvers had certain abilities that poor problem solvers lacked, such as the following:

1. An ability to formalize mathematical material, to isolate form from content, to abstract oneself from concrete numerical relationships and spatial forms, and to operate with formal structure — with structures of relationships and connections.
2. An ability to generalize mathematical material, to detect what is of chief importance, abstracting oneself from the irrelevant, and to see what is common in what is externally different.
3. An ability to shorten the reasoning process, to think in curtailed structures.
4. A mathematical memory. It can be assumed that its characteristics also arise from the specific features of the mathematical sciences, that this is a memory for generalizations, formalized structures, and logical schemes (pp. 87-88).
Silver (1979) concurred with Krutetskii's conclusion that good problem solvers are superior to poor problem solvers in their ability to perceive mathematical structure. The positive correlation between students' problem solving ability and perception of mathematical structure was significant even when Silver controlled the effects of verbal and nonverbal IQ, mathematical concepts knowledge, and computational ability. Foster (1972) and Chartoff (1976) also found a relationship between students' problem solving ability and ability to generalize. In addition, researchers have shown a positive correlation of other mathematical thinking abilities and students' problem solving ability (e.g. Schoenfeld, 1982; Dodson, 1972). Analysis of the relationship of programming and mathematical generalization may provide insight into programming's effects on problem solving ability.

3. **Situation Variables**

The third category of Kilpatrick's problem solving research variables includes features of the problem solving environment external to the problem and problem solver. Instructional factors comprise the most important class of factors within this category (Lester, 1980).

There have been several dissertation studies investigating the apparent similarities between problem solving heuristics and programming techniques (e.g. Hersberger, 1983; Wells, 1981; Foster, 1972). According to Davis (1983), "The fastest road to achievement in mathematics" is computer programming since "computer programming is taught as problem solving." Research literature for the most part
seems to indicate a weak positive relationship between problem solving ability and programming instruction (Hersberger, 1983). For example, Johnson and Harding (1979) studied the effect of computer programming on the problem solving ability of Cambridge University students. The students who completed a computer science elective did significantly better on the annual Tripos exams. The researchers felt these results were relevant because the students had no prior programming experience and the Tripos exams are generally accepted as realistic and consistent assessments of mathematical problem solving ability. However, Cambridge students must be regarded as mathematically gifted. Hatfield and Kieren (1972) also found the high school students (seventh and eleventh grades) involved in the computer programming experience significantly outperformed the control group on a test of non-routine problems that required "careful analysis, reasoning, and insightful solutions" (p. 106). In neither study, unlike the studies of Hersberger (1983) and Wells (1981), was an attempt made to observe the problem solving processes used by the subjects. Blume (1984) states that

Knowledge of the extent of the influence and the means by which programming instruction may influence aspects of problem solving are important considerations if instruction is to be designed that maximizes its contribution to problem solving (p. 2).

Theories and hypotheses for problem solving instruction are often based on the works of Polya (1957). One of the key ideas in Polya's work is the use of problem solving heuristics. Polya described a "heuristic" as a planned action or series of actions performed to assist in the discovery of a solution to a problem. He suggests that
"heuristical reasoning is reasoning not regarded as final and strict but as provisional and plausible only, whose purpose is to discover the solution of the present problem" (p.113). Common heuristics are working a simpler problem, drawing a diagram, and looking for a pattern.

Using Polya's problem solving methodology as a guideline for her own clinical research as well as to review the literature, Wells (1981) concluded there is initial evidence the same processes are used in computer programming and in problem solving. Common heuristics included use of subgoals, looking back, trial and error, and looking for a pattern.
GENERALIZATION

While generalization is only one of the mental processes that correlate positively with mathematical aptitude, many mathematicians feel it is an extremely important attribute of mathematical maturity (Shumway, 1968; Krutetskii, 1976). Krutetskii states that "abstractions and generalizations constitute the essence of mathematics and mathematical thinking" (p.86). He further states that "an ability to generalize mathematical material, to detect what is of chief importance, to [ignore] the irrelevant, and to see what is common in what is externally different" is a basic characteristic of mathematical thought (p.87). Mathematics educators draw a distinction between abstraction and generalization (Dienes, 1961). Abstraction is the process of drawing from a number of different situations something which is common to them all. For example, a young child views various distinct pairs of objects and observes that the essential common property of all such pairs is the natural number two. While the process of abstraction leads the learner from particular elements to a common class, the process of generalization leads from classes to classes. In Figure 2, Skemp (1971) describes the mental process involved in generalization (pp. 60-61).
From a set of examples, a general method is derived.

Which can be applied to other examples of the same kind.

The method is next formulated explicitly, considered as an entity in itself, and its structure analysed.

This structure is used to invent ways of using the same method for examples of a new kind. The original examples are included in the enlarged field of application of the method.

Figure 2. Skemp's Description of Mathematical Generalization

Dienes identifies two types of generalization, primitive and mathematical. Primitive generalization is the passing from one class to another where the new class includes the former as a subclass. For example, if a child notices commutativity holds for several pairs of natural numbers, then s/he has formed the concept for that restricted subset of the natural numbers. When s/he realizes that the principle is true for any pair of natural numbers, s/he has generalized from the class of his/her actual experience to the class of natural numbers.
Finally, according to Dienes, mathematical generalization is the process of extending a universe to a class that contains an isomorphic image (in relation to all relevant properties) of the restricted class. For instance, the extension of a universe from the natural numbers to the integers. The natural numbers exhibit the same properties as their isomorphic image, the positive integers.

Advocates of computers in mathematics education often cite the apparent relationship between certain mathematical thought processes; in particular, generalization, and the processes involved in computer programming (e.g. Hatfield, 1973; Feurzig and Papert, 1968). For example, the process of mathematical generalization is implicit in Feurzig and Papert's (1968) description of programming as a "constructive problem solving process."

A solution to a problem is built according to a preconceived, but modifiable, plan, out of parts which might also be used in building other solutions to the same or other problems (p. 12).

Similarly, Hatfield (1971) perceives the process of programming as "successive approximation" since first a person tends to solve the problem for a restricted set of data and then extends and modifies the program for a larger universe. It can be argued that programming places a person in an environment where one is encouraged to extend, revise, and refine efforts to a more general algorithm aimed at processing an entire classes of problems. In other words, through programming a person is involved in mathematical generalization.

Unfortunately, there exists little empirical evidence supporting the claim of a positive relation between computer programming and
mathematical generalization. Yet, as DeCorte states, "there is also no counterevidence" to refute the hypothesis (1984, p. 1). Recalling the correlation between problem solving ability and mathematical generalization, research supporting the connection between programming and this cognitive process could serve as justification for the implementation of microcomputers into the mathematics curriculum.
In an effort to verify the effects of computer instruction on mathematics achievement, the relationship of computer programming ability to mathematical generalization and understanding of variables was studied. Two different instructional methods were employed as a means of identifying factors which influence this relationship.

OVERVIEW

Below is a brief summary of the experimental procedures used. Pertinent details and further information can be found in the following sections.

A. Treatment - Two different instructional methodologies were used - the Wholistic and Elemental approaches. The Wholistic approach (Treatment W) began instruction at the whole program level. The focus was on mathematically relevant problems with commands introduced only as needed to solve the problem. On the other hand, the Elemental approach (Treatment E) focused on the individual BASIC commands and how they function. Instruction began with
the simplest commands and proceeded stepwise until the student is capable of programming complex problems.

B. Subjects - In order to study the two relationships cited above, four homogenous groups were formed. Two groups acted as control groups (Groups C1 and C2); the second control group was used to measure pretest effects. The students of the other two groups were taught programming by two different methods (Groups W and E). Sixth grade mathematics classes from two elementary schools participated. Both schools are located in a small midwestern city near a large university and have similar student populations with respect to age, sex, SES, IQ, mathematical background and programming ability. The students of one school were randomly assigned to Groups C1 and C2 and the others to Groups W and E.

C. Procedure - For 6 weeks, each treatment group was given 60-90 minutes of programming instruction 2 or 3 times a week. Overall, the students were involved in approximately 20 hours of programming instruction. Some students spent additional time (e.g. at the end of class, during recess) working on the computers which were located in their mathematics classroom.
D. **Instruments** - Pre- and posttests assessing programming ability (PROG), mathematical generalization (GEN), and understanding of variables (VAR) were administered to subjects in the treatment groups and half the students in the control group. The other half of the control group took non-relevant mathematics tests (i.e. a placebo) as pretests. The testing took a total of 50 minutes at the beginning and end of the treatment. Five students from each treatment group were selected for interviews of approximately 40 minutes. To aid in analysis of the results, additional information such as CTBS scores, age, and sex were collected.

**PILOT STUDY**

The pilot study can be divided into two phases — development of the instruments and development of the programming activities for the Wholistic Treatment. The instruments and the activities were field tested and revised several times. The field test groups varied in age, mathematical background, and socio-economic status. After each revision, the instruments and/or activities were reviewed by a panel of experts which included mathematics educators, mathematicians, computer scientists, and elementary school teachers. The panel provided insightful comments with regard to topic suggestions, reading level,
programming suggestions, and instructional methods. A complete description of the pilot study along with sample items and descriptive statistics can be found in Appendix B.

TREATMENT

In the elementary grades, often the sole goal of programming instruction is the development of programming skills and not the application of these skills in mathematical problem solving. The beginning focus is on the elementary commands of the programming language, PRINT and LET for example, rather than the structural commands such as FOR/NEXT. As an alternate method, classroom activities have been designed that take a discovery/problem solving approach to computer programming (Treatment W). According to Damarin (1982), "The teaching of general problem solving requires examination of both real world phenomena and abstract symbolic systems" (p.11). Figure 3 illustrates the global relationship between these structures.

Figure 3. Model of the relationship of real world problems and mathematical systems
In accordance with this model, each computer activity of treatment W posed a mathematically relevant problem and only the commands needed to solve that problem were introduced (see Appendix C). Hamrock (1974) referred to this method of programming instruction as the "whole program" approach.

The whole program approach begins instruction at the whole program level. The students learn the fundamentals of the computer language through the programs themselves by a discovery type sequence of instruction. The emphasis is on writing programs and working through previously written programs to develop a capacity of writing programs for computer solution of complex problems. Topics are covered as needed for the completion of the programming assignment, rather than by increasing difficulty or a pre-determined logic sequence. (p. 32)

Johnson (1983) is also an advocate of the wholistic method of programming instruction. He states that programming is best learned by following a sequence of activities which involve the following:

A. Using "blocks of code" (short programs) to do significant and/or interesting tasks (children will see the "words" working).
B. Modifying a short program to do a new but related task.
C. And finally designing or preparing a new program to solve a problem (p. v).

As stated earlier, mathematical generalization is the ability to detect structure and similarity in externally dissimilar situations. In order to further emphasize generalization, the students in Treatment W were encouraged to write efficient programs and to look for similarities between programs. Specifically, the following strategies were stressed.
1. To look for different approaches to the same problem.
2. To make a programming algorithm more general.
3. To apply an algorithm to different situations.

Mathematical or programming concepts which have an underlying structure evident in the programmed solutions were used as the basis of the Wholistic Treatment computer activities. These topics -- counting, nested loops, sorting, and drawing figures -- were "re-occurring themes" throughout the treatment. For example, the first lesson involved "teaching the computer to count." The students learned to write simple programs such as Figure 4.

```
10 FOR X = 1 TO 10
20 PRINT X
30 NEXT X
```

Figure 4. Sample introductory program for Treatment W

A few class sessions later, the students were asked to write a program simulating a countdown for the space shuttle (see Figure 5). Clearly the two programs are structurally similar.

```
10 FOR X = 100 TO 0 STEP -1
20 PRINT X
30 NEXT X
40 PRINT "BLAST OFF!!!"
```

Figure 5. Sample follow-up program for Treatment W
The Elemental or command-oriented group (Group E) was involved in the more "traditional" approach to computer programming. The stress was on the fundamental elements of programming - BASIC commands - and how they function. Hamrock (1974) referred to this method as the "structural" approach to computer programming instruction and described it as follows:

The structural approach builds the programming language in a hierarchy of topics, much like Gagne’s building blocks of instruction. The instruction begins with the simplest topics and ideas, and adds to these until the student is capable of programming complex problems for computer solution. The emphasis is on writing small segments of programs properly, and working with each language topic prior to any emphasis on the writing and running of whole programs (p.31).

The book used by Group E, Basic_Discoveries (Malone and Johnson, 1981), contains activities that allow the students to explore the different commands by entering and running simple programs. It is the goal of the authors to...

give students a thorough understanding of the rudiments of programming. The activities of the book provide a firm foundation for learning more advanced programming techniques. It is the intent for all students using this book to learn to use a computer as a versatile problem solving tool with which they can solve non-trivial problems that are of interest to them (p. v).

While the programs in the text are carefully constructed examples of the various commands, they are usually devoid of any meaningful mathematical content (see Appendix D).

The same programming commands were covered in both treatments but not necessarily in the same order. The commands that were covered are as follows: PRINT, LET, FOR/NEXT, STEP, NEW, END, LIST, RUN, TAB,
INPUT, IF/THEN (optional), RND, and INT. It should be noted, the students in Treatment E wrote some of the same programs as the students in Treatment W (e.g. the simulation of a countdown) but only after studying and practicing each command needed to write the program.

The two instructional methods also varied as to the cognitive demands placed on the respective subjects. Using Bloom's (1956) taxonomy of educational objectives, the researcher categorized the intellectual abilities and skills required of the subjects of the two instructional treatments. The activities of Treatment W resembled problem solving tasks and hence required the use of analysis and evaluation to deal with a given task. On the other hand, comprehension and application were the cognitive behaviors most likely employed by the students in treatment E. The following situations are illustrations of the different cognitive behavior objectives of the two treatments.

Recall, the subjects of Treatment W were presented with the task of writing a program to simulate a countdown for the space shuttle (see Figure 5). The students were encouraged to compare this task to previously written programs and to look for similarities. Earlier, the students had written programs that "taught the computer to count" (see Figure 4). To write the countdown program, a student would need to analyze the present task and identify the similarities between it and the previously learned "counting" algorithm. After analyzing the task, the student evaluated the underlying structure (e.g. the FOR/NEXT loop used for counting) in order to choose the appropriate algorithm to apply. Evaluation or "judgements about the extent to which material
and methods satisfy criteria" is the next cognitive mode required of the student (Bloom, p. 207). While some evaluation may have occurred during program development, it was more likely to occur during and after program execution. Since the students in Treatment W were encouraged to write efficient programs that were applicable to a broad range of tasks then during the debugging and refining process, these students were constantly evaluating the program and comparing alternatives.

For the students of Treatment E, comprehension and application were the intellectual behaviors used for dealing with programming problems. A typical lesson began with the instructor defining a programming command and illustrating the command with a simple program. For example, Figure 6 is the program used to introduce the INPUT command:

```
10 PRINT "TYPE A NUMBER"
20 INPUT N
30 PRINT N, 2*N, N*N
40 END
```

Figure 6. Sample introductory program for Treatment E

Next the students completed a worksheet which required them to enter programs similar to Figure 6 into the computer and record the output (see Appendix D, Activity 20). It is the intent of the authors that the students comprehended the INPUT command after completing the worksheet. According to Bloom, comprehension is the lowest level of understanding, the student "knows what is being communicated and can
make use of the material or idea being communicated without necessarily relating it to other material or seeing its fullest implications" (p. 204). The next programming task for these students was a direct application of this particular command. Figure 7 is one of the follow-up activity for the INPUT command.

Using INPUT statements, write a program that makes the computer print the area of any triangle, given the base and height of the triangle (area = 1/2 * base * height).

Figure 7. Sample follow-up activity for Treatment E

When assigned the follow-up activity, the students were instructed to use the programs of Activity 20 as models of the INPUT command.

The activities of Treatment W emphasized a higher level of cognitive behavior. It was hypothesized this higher level of thought would result in a higher level of programming. Furthermore, because of the emphasis in Treatment W on underlying structural similarities and not individual programming commands, it was also hypothesized the group receiving Treatment W would have the greatest increase in mathematical generalization and the greatest correlation between generalization and programming. Table 1 summarizes the differences between the two instructional approaches.
Table 1

<table>
<thead>
<tr>
<th>Differences of Programming Instruction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Treatment</td>
</tr>
<tr>
<td>W</td>
</tr>
<tr>
<td>E</td>
</tr>
<tr>
<td>Source</td>
</tr>
<tr>
<td>Focus</td>
</tr>
<tr>
<td>Cognitive Level</td>
</tr>
<tr>
<td>Presentation of commands</td>
</tr>
</tbody>
</table>

SUBJECTS

There were several reasons for choosing sixth grade students as subjects. One reason was the prerequisite mathematical knowledge required in some of the activities of the two treatments. Prerequisite knowledge included application of multiplication (e.g. calculating area of rectangle) and the concept of factors. Most sixth graders should have comprehension of these concepts.

A second reason is the level of general reasoning ability of sixth grade students. According to Piaget’s theory (Inhelder and Piaget,
1958), twelve year olds are in a transition from concrete operational to formal operational thought. At this age level, adolescents are capable of formal reasoning but still require observation and manipulation of real world experience. Learning experiences built on action and interaction with the real world have the best chance of being incorporated into the adolescent's conceptual orientation (Piaget, 1976). The computer can provide students the various experiences which can nurture flexibility and diversity of thinking (Damarin, 1982).

The school which was directly involved in the treatment had 3 sixth grade mathematics classes taught by a single teacher. These classes were not grouped by ability. Because of scheduling conflicts, one of the three class was not used. The students of the other two classes studied BASIC programming using nine microcomputers, seven of which were provided by the researcher, located in the mathematics classroom. The students were randomly assigned to two different treatments (Treatment W and E) with the stipulation that treatment groups have approximately the same proportion of girls (see Table 2).

Table 2
Summary of Treatment Groups by Sex and Class

<table>
<thead>
<tr>
<th>Class</th>
<th>Group</th>
<th>Sex</th>
<th>1</th>
<th>2</th>
<th>Subtotal</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>M</td>
<td>5</td>
<td>7</td>
<td></td>
<td>12</td>
<td>26</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>8</td>
<td>6</td>
<td></td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>M</td>
<td>9</td>
<td>5</td>
<td></td>
<td>14</td>
<td>29</td>
</tr>
<tr>
<td></td>
<td>F</td>
<td>7</td>
<td>8</td>
<td></td>
<td>15</td>
<td></td>
</tr>
</tbody>
</table>
The school that served as the control, also has 3 sixth grade classes. However these students were grouped by ability. Thus to have control groups that were comparable in ability to the treatment groups, all three classes were used. The students of each class were randomly assigned to one of the two control groups (Groups C1 and C2), again with the stipulation that the groups contain approximately the same proportion of girls. See Table 3 for a breakdown of the subjects in the two control groups.

Table 3
Summary of Control Groups by Sex and Class

<table>
<thead>
<tr>
<th>Class</th>
<th>Group</th>
<th>Sex</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>Subtotal</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C1</td>
<td>M</td>
<td>3</td>
<td>6</td>
<td>4</td>
<td>13</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>15</td>
<td></td>
</tr>
<tr>
<td></td>
<td>C2</td>
<td>M</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>11</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>F</td>
<td>2</td>
<td>5</td>
<td>5</td>
<td>12</td>
<td></td>
</tr>
</tbody>
</table>

While the researcher would have preferred subjects to be naive programmers, the sixth graders of both schools had studied in an informal three week course in the programming language Logo. Once a week, the computer coordinator would conduct a 60 minute lecture/discussion class in place of mathematics. In addition, each student spent approximately thirty minutes per week on the computer. Logo is a programming language designed for use by young children. Based on geometric principles, the user can draw relatively complex designs with a few simple commands. Many of the Logo activities were taken from Geometry Problems for Logo Discoveries (1984) and the following are the topics covered (Troutner, 1984):
1. Fundamental commands - FD, BK, RT, LT, PU, PD
2. How to draw your own initials
3. REPEAT command
4. The law of 360°
5. SET X,Y command
6. Given a picture, write the commands to draw it
7. Draw your own picture

When using Logo, the students always worked in the direct mode since procedures were not taught. Furthermore, since the students only had access to one computer, they were required to do all planning before using the computer. Being exposed to Logo, the students may have gained an intuitive understanding of how a computer operates. However, because of the limited nature of the instruction, the researcher did not expect this experience would significantly affect the results of the study.

PROCEDURES

For 6 weeks, each treatment group was given 60-90 minutes of programming instruction 2 or 3 times a week. Overall, the students were involved in approximately 20 hours of programming instruction. These instructional sessions involved approximately 30 minutes of class discussion and 60 minutes of computer time. During a typical session, each pair of students would be able to work at a computer twice, fifteen minutes each time. The students were also able to work on the computers after school and/or during recess. Several students from each of the treatment groups would take advantage of this additional
computer time once or twice a week. The researcher was frequently available during the additional computer time to answer students' questions and observe student progress.

During each instructional period, the researcher taught Group W and the mathematics teacher taught Group E. Although the two groups were receiving instruction simultaneously, the physical arrangement of the room was such that they remained separated even while working on the computers (see Figure 8). In addition to two Apple IIe microcomputers seven more computers were placed in the mathematics classroom, namely, three Commodore PET and four TI 99/4A. As is evident in Figure 8, Group W had access to 5 computers while Group E had use of only four.

![Figure 8. Seating and computer arrangement of the treatment classroom](image-url)
Since programming concepts can be abstract for this age level, the students needed time to assimilate the BASIC commands and reach a measurable level of generalization. Some researchers believe that in such a study, students need at least 50 hours of instruction (DeCorte, 1984; Milojkovic, 1983). However, confronted by the normal limitations found in a school environment, only 20 hours of instruction were possible.

INSTRUMENTS

In order to test the major hypotheses, three instruments were developed to measure the following processes:

1. Mathematical Generalization (GEN)
2. Programming ability (PROG)
3. Understanding of Variables (VAR)

Each instrument was administered as a pretest and posttest. The students of all three treatments were given the tests. About 20 minutes was needed to administer the GEN and PROG tests and 10 minutes for the VAR test. Thus testing took a total of 50 minutes at the beginning and end of the treatment. Copies of the instruments can be found in Appendix A.
1. Generalization Instrument

The Generalization instrument is based on Diene's (1961) definition of mathematical generalization. Mathematical generalization is a mental process; namely, the ability to detect similarities in mathematical situations that appear externally different. For any given problem, a student is able to achieve some level of generalization. While the levels of generalization should be thought of as a continuum, in order to develop an instrument, the following behaviors, proposed by Doll (1983), were used to approximate specific levels.

1. See a pattern
2. Utilize a pattern
3. Extend a pattern
4. Make a generalization

Being that generalization is a cognitive process employed during problem solving, it follows the intellectual demands placed on students in the act of generalization are those associated with problem solving. As shown in Table 4, Doll's levels of generalization correspond with specific intellectual behaviors of Bloom's (1956) taxonomy.

Table 4
Correspondence of Doll's Generalization Levels to Bloom's Taxonomy

<table>
<thead>
<tr>
<th>Generalization Level</th>
<th>Cognitive Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>See a pattern</td>
<td>Analysis</td>
</tr>
<tr>
<td>Utilize a pattern</td>
<td>Application</td>
</tr>
<tr>
<td>Extend a pattern</td>
<td>Analysis</td>
</tr>
<tr>
<td>Make a generalization</td>
<td>Evaluation</td>
</tr>
</tbody>
</table>
In order to "see a pattern," a student must be able to analyze the set of examples and identify similarities. While "using a pattern" implies the direct application of that pattern to similar tasks, "extending a pattern" is the application of that pattern to new tasks which appear dissimilar to the original tasks. Thus, once again, the student must analyze the new set of examples and identify the similarities in underlying structure with the old set. Finally, to "make a generalization" a student must evaluate all of the examples to judge the extent of which the examples satisfy the criteria of the proposed generalization.

The items were adapted from instruments used in related research (Krutetskii, 1976; Silver, 1979; Burger, 1980). The Generalization instrument consisted of four subtests, each involved a different mathematical concept. The four concepts were combinations, exponents, number patterns, and geometrical shapes. The items from subtest 1 are listed in Figure 9 (see p. 50).

The items of each subtest are ordered and weighted so that hypothetically the number of items successfully completed indicates students' level of generalization. In other words, for items 1-5, students must see the pattern of multiplication and for items 6 and 7 they must use this pattern (see Figure 9, p. 50). Extension of the pattern is required for item 8 and finally for the last item the student is asked to make a generalization.
**EXAMPLE:** How many different pairs can you make from these two sets (0,1,2) and \{a,b,c\}?

**SOLUTION:** List the pairs and count.

0, a  0, b  0, c  
1, a  1, b  1, c  
2, a  2, b  2, c

**ANSWER:** 9 pairs

**NOTE:** The 3 elements of the first set are 0, 1, 2. The 3 elements of the second set are a, b, c.

1. How many different pairs can you make from these two sets (0,1,2) and \{d,e,f\}?  
2. How many different pairs can you make from these two sets \(x,y,z\) and \(\{a\#,*\}\)?  
3. How many different pairs can you make from these sets \{1,2,3,4\} and \{a,b,c,d\}?  
4. How many different pairs can you make from these sets \{w,x,y,z\} and \{q,e,r,t\}?  
5. How many different pairs can you make from \{0,1,2,3,4\} and \{a,b,c\}?  
6. How many different pairs can you make from a set A which has 6 elements and a set B which has 3 elements?  
7. How many different pairs can you make from a set A which has 13 elements and a set B which has 0 elements?  
8. How many different outfits can you make if you have 8 shirts and 9 pants?  
9. Think about how you solved problems 1-8. Below are some ways for solving problems of this type. Some of these methods do not work. Some work but take awhile. Pick the method that you feel is the quickest and most general method that works for all these problems. If you pick D, write your answer on the line on the answer sheet.

A. Multiply the number of elements in the first set by the number of elements in the second set.  
B. List all of the pairs and count them.  
C. If each set has 3 elements then there are 9 pairs. If each has 4 elements then there are 16 pairs. If each has 5 elements then there are 25 pairs, and so on.  
D. None of the above. The way to find all the pairs is to...

---

Figure 9. Subtest 1 of the Generalization instrument
In summary, Table 5 lists the items of the Generalization test according to Doll's behavioral objectives and mathematical content.

Table 5
Generalization Test Items by Behavioral Objectives and Mathematical Content

<table>
<thead>
<tr>
<th>Objective</th>
<th>Mathematical Content</th>
<th>See a Pattern</th>
<th>Utilize a Pattern</th>
<th>Extend a Pattern</th>
<th>Make a Generalization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Combinations</td>
<td>1,2,3,4,5</td>
<td>6,7</td>
<td>8</td>
<td>9</td>
<td></td>
</tr>
<tr>
<td>Exponents</td>
<td>10,11,12,13</td>
<td>--</td>
<td>--</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>Num. Patterns</td>
<td>15</td>
<td>--</td>
<td>17</td>
<td>16</td>
<td></td>
</tr>
<tr>
<td>Geom. shapes</td>
<td>18</td>
<td>--</td>
<td>20</td>
<td>19</td>
<td></td>
</tr>
</tbody>
</table>

Note, although there were 22 test items, there are only 20 items listed in Table 5. For scoring purposes, test items 16 and 17 were combined and test items 20 and 21 were combined. Thus, the items 16 through 22 were renumbered. In Table 5, item 16 refers to test items 16 and 17 and item 19 refers to test items 20 and 21. Furthermore, items 17, 18, and 20 are, respectively, test items 18, 19, and 22.

2. Programming Instrument

The ability to program encompasses several skills and processes. Johnson (1980) developed a series of behavioral objectives to be used as a guideline for test construction in a computer literacy course. Below are Johnson's objectives pertaining to computer programming:
The student should be able to accomplish objectives 1.2-2.5 when the algorithm is expressed as a set of English language instructions and is in the form of a computer program.

1.0 Recognize the definition of "algorithm"
1.2 Follow and give the correct output for a single algorithm.
1.3 Given a simple algorithm, explain what it accomplishes (i.e., interpret and generalize).
2.1 Modify a simple algorithm to accomplish a new, but related, task.
2.2 Detect logic errors in an algorithm.
2.3 Correct errors in an improperly functioning algorithm.
2.4 Develop an algorithm for solving a specific problem.
2.5 Develop an algorithm that can be used to solve a set of similar problems (p. 93).

In addition to these objectives, several other factors influenced the development of the Programming instrument; such as the following:

1. Item format
2. BASIC programming commands
3. Cognitive demand

The test items were adopted from several sources and varied in format from fill-in-the-blank to writing programs. Since the tasks ranged from identifying definitions of commands to debugging syntax and logical errors, the student was involved in a range of cognitive demands. Based on Bloom's (1956) taxonomy of education objectives, the cognitive demands placed on the subjects varied from simple recall of terms to analysis and evaluation. Finally, a total of 13 BASIC commands were used on the test plus various punctuation marks (comma,
With such a variety of constraints, each item was assessing a unique aspect of programming ability.

3. Understanding of Variables Instrument

Like the Generalization instrument, the items on the Understanding of Variables instrument were adopted from several other studies (Clement, 1979; Wagner, 1981; Milner, 1972). Understanding the concept of variables involves the ability to recognize a variable (i.e. knowing the relevant attributes) and simple applications. Specifically, the instrument covers three behavioral objectives:

1. Evaluate simple algebraic expressions.
2. Translate verbal statements to symbolic equations.
3. Recognize all possible equations for a verbal statement.

For the Variable instruments, items of different complexity level were selected. The complexity level of an item is based on the minimum number of steps required to solve that item. As illustrated in Figure 10, problem V02 is a "one-step" problem while problem V03 is a "two-step".
Problems:

<table>
<thead>
<tr>
<th>Value of variables</th>
<th>Expression</th>
<th>Value of expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>2. f = 100, e = 4</td>
<td>f+3</td>
<td></td>
</tr>
<tr>
<td>3. g = 9</td>
<td>g x (g+1)</td>
<td></td>
</tr>
</tbody>
</table>

Solutions:

2. f+3 = 100+3 =103 (step 1 - add)
3. g x (g+1) = 9 x (9+1) = 9 x 10 = 90 (step 2 - multiply)

Figure 10. Examples of one-step and two-step problems from the Variable instrument.

The steps of the solution process need not necessarily be arithmetical computation. Problems V06-V08 are also two-step problems since the student must first translate the written words into algebraic symbols and then write a balanced equation that corresponds to the verbal statement. This is best illustrated by examining an incorrect solution to problem V07 (See Figure 11). The students were asked to write an equation for the sentence below.

7. Mike has 3 less books than Kim.

Right Answer: M = K - 3
Wrong Answer: M - 3 = K

Figure 11. Two-step problem which does not involve arithmetical computation.
A student who wrote the incorrect equation was able to translate the written words into algebraic symbols but was unable to write a balanced equation that corresponded to the verbal sentence. The items of the Variable test are listed by behavioral objective and minimum steps to solution in Table 6.

Table 6
Variable Test Items by Behavioral Objectives and Complexity Level

<table>
<thead>
<tr>
<th>Behavioral Objective</th>
<th>Steps</th>
<th>Evaluate</th>
<th>Translate</th>
<th>Recognize</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>V01, V02</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>2</td>
<td>V03, V04</td>
<td>V06-V08</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>&gt;3</td>
<td>V05</td>
<td>--</td>
<td>V09</td>
<td></td>
</tr>
</tbody>
</table>

4. Interviews

In addition to the paper and pencil instruments, further information was obtained by interviews of selected student. According to Burger (1984), interviews are essential for theoretical model building that involve cognitive processes, such as generalization, in children since through interviews the educational researchers can assess the childrens' characteristics and attempt to relate these characteristics to the cognitive process. Specifically, the interviews provided an opportunity to observe students' methods of generalization in a problem solving situation and to look for differences between students of the two treatment groups.

To obtain a sufficient amount of usable data, Burger (1984) recommends the following procedures for selection of interview
subjects.

1. Select only a few children but try to get a good mixture with respect to mathematical ability and sex.

2. Do not make random selections. Have someone familiar with the children make the selection with the objective of choosing highly verbal children.

Accordingly, the case study subjects were chosen by the mathematics teacher who was instructed to select 10 students, 5 from each treatment group, with an approximately equal numbers of boys and girls. These students were chosen for their willingness and ability to communicate and varied in mathematical ability and previous programming experience. The one-on-one interviews lasted approximately 40 minutes each and took place either after school or during lunch. The interview questions were similar to the Generalization instrument and can be found in Appendix D.

5. Observations

For each treatment group, the instruction and student programming sessions were audiotaped with tape recorders placed by the computers. There was one tape recorder at each table for a total of two per treatment group. Occasionally during the treatment, the researcher would review these tapes to assess the instructors compliance with the intentions of the two treatment. At the end of the treatments, the tapes were transcribed and analyzed, with the intent of identifying factors which support and/or clarify the statistical results.
To supplement the observations, the instructors of Groups W and E rated their respective students' persistence, motivation, and understanding on a five point scale. Understanding refers to the student's understanding of the BASIC computer language and programming. A student's rating was based on the instructor's observations of that student's behavior during the computer programming instruction. The instructor was also asked to list pairs of students that worked together exceptionally well or poorly.

6. Non-relevant Mathematics Instruments

To measure pretest effect, half of the control subjects took non-relevant mathematics tests. The researcher developed two instruments (NON1 and NON2) using items from Indiana Mathematics League 1983 Mathematics contest. NON1 was approximately the same length as the Generalization (GEN) and Programming (PROG) instrument while NON2 was shorter and equivalent in length to the Variable (VAR) instrument. For all subjects, testing involved two days. Table 7 illustrates the pretesting procedures for the control groups. Both groups took the GEN, PROG, and VAR posttests.

Table 7
Pretesting Procedures for the Control Groups

<table>
<thead>
<tr>
<th>Group</th>
<th>Day</th>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td>One</td>
<td>NON1</td>
<td>PROG</td>
<td></td>
</tr>
<tr>
<td>Two</td>
<td>GEN</td>
<td>NON1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>VAR</td>
<td>NON2</td>
<td></td>
</tr>
</tbody>
</table>
Finally, Table 8 is a summary of the treatment and testing procedures.

<table>
<thead>
<tr>
<th>Group</th>
<th>Treatment and Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>O_{bpv} -- X -- O_{bpv}</td>
</tr>
<tr>
<td>B</td>
<td>O_{bpv} -- X -- O_{bpv}</td>
</tr>
<tr>
<td>C1</td>
<td>O_{av} ------- O_{av}</td>
</tr>
<tr>
<td>C2</td>
<td>O_{p} ------- O_{av}</td>
</tr>
</tbody>
</table>

Note: Group C1 took a non-relevant mathematics test in place of the Programming instrument and group C2 was administered non-relevant exams in place of the Generalization and Variable instruments.

STATISTICAL ANALYSIS

1. Experimental Design

The experimental design is illustrated in Table 7. To measure pretest effect, half of the control subjects took non-relevant mathematics tests instead of the three research instruments. Group C1 took a non-relevant mathematics test in place of the programming instrument and group C2 was given non-relevant exams instead of the generalization and variable instruments. Since there was not a statistically significant pretest effect, subjects were randomly
selected from groups C1 and C2 to form group C. Group C was to be the control group in all further statistical analysis. Table 9 illustrates this modified experimental design.

Table 9

<table>
<thead>
<tr>
<th>Group</th>
<th>Modified Experimental Design</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X ----- O</td>
</tr>
<tr>
<td>B</td>
<td>X ----- O</td>
</tr>
<tr>
<td>C</td>
<td>----- O</td>
</tr>
</tbody>
</table>

2. Research Questions

1. Will the method of programming instruction significantly effect a student's...
   a) mathematical generalization?
   b) programming ability?
   c) understanding of variables?

2. What is the effect of computer programming instruction on a student's...
   a) mathematical generalization?
   b) programming ability?
   c) understanding of variables?

3. Will any of the following factors be related to a student’s mathematical generalization, programming ability, or understanding of variables?
   a) Sex
   b) Mathematical ability
   c) IQ
   d) Previous programming ability
   e) Microcomputer in the home
   f) Motivation
   g) Persistence

4. Is there a pretest effect for any of the three instruments?
3. **Statistical Model**

In order to address the first two research questions, the statistical model used was a multivariate analysis of variance (MANOVA). Since the posttests (GEN P0, PROG P0, and VAR P0) measuring the effects of the three dependent variables were administered simultaneously, it was more appropriate to use a multivariate analysis instead of three analyses of variance (Campbell and Stanley, 1966; Morrison, 1967). Multivariate analysis essentially tests the null hypothesis:

$$H_0: \mathbf{M} = \mathbf{M}_W = \mathbf{M}_E = \mathbf{M}_C$$

where $\mathbf{M}_i = [\mathbf{M}_{1,i}, \mathbf{M}_{2,i}, \mathbf{M}_{3,i}]$, $i = W, E, or C$. In other words, it is a test of the hypothesis that the three groups (treatment groups W and E and control group C) arose from populations with a common mean vector. Multivariate analysis of variance is not sufficient to determine specific effects of the three dependent variables. Furthermore, as in the univariate analysis of variance, rejection of the null hypothesis does not indicate which treatments or treatment combinations are significantly different. Multiple comparisons tests were used to address these two issues. To determine which of the individual dependent variables may have led to the rejection, the SPSS program, MANOVA, performs three separate ANOVA's using approximated mean squares. These procedures tested the following null hypotheses:
where $\bar{M}_o, \bar{M}_p, \bar{M}_v$ are the individual components of the mean vector $M_I$, $I = W, E, C$. These statistics could be considered since the multivariate results were significant.

If any of these hypotheses are rejected, the specific effects of the treatments can be considered. The first research question was addressed by testing the following null hypotheses using multiple comparison tests.

$$H_{o1G}: M_0 = M_{0,W} = M_{0,E} = M_{0,C}$$
$$H_{o1P}: M_p = M_{p,W} = M_{p,E} = M_{p,C}$$
$$H_{o1V}: M_v = M_{v,W} = M_{v,E} = M_{v,C}$$

The second research question was addressed by using compound not pairwise comparisons. The following three null hypotheses were considered.

$$H_{o2G}: M_{o,ave} = M_{o,C}$$
$$H_{o2P}: M_{p,ave} = M_{p,C}$$
$$H_{o2V}: M_{v,ave} = M_{v,C}$$

where $M_{i,ave}$ refers to the average mean of $M_i,W$ and $M_i,E$, $I = G, P, V$. Dunn's test was used for the post hoc analysis since it could be used for pairwise and compound comparisons and when the treatment groups have unequal n's. Furthermore, Dunn's test is also the most conservative of all such procedures.
Since there is little research of the relationship of computer programming to various mathematical cognition processes, this study was partially exploratory in nature. It was feasible to believe other factors such as sex, a microcomputer in the home, or previous programming experience may have significantly influenced the relationship. Thus the correlation between PROG PO, GEN PO, and VAR PO and the various factors obtained from the background questionnaires were calculated. If any of these variables were found to have a significant correlation, they would be incorporated into the statistical model. While the researcher was interested in examining the relationship between the dependent variables and each independent variable, simple correlations were not appropriate because the other independent variables may have confounded the results. Instead, multiple regression was used to control possible confounding factors in order to evaluate the contribution of a specific variable or set of variables. The following null hypotheses were tested.

\[ H_0^G: \beta_0 = 0 \]
\[ H_0^P: \beta_P = 0 \]
\[ H_0^V: \beta_V = 0 \]

Where \( \beta_i = [\beta_{i,b}, \beta_{i,c}, \beta_{i,m}, \beta_{i,s}, \beta_{i,t}, \beta_{i,u}] \), \( i = G, P, \) or \( V \), and \( b, ..., g \) refer to the factors in research question 3. Actually since the researcher was only interested in determining the degree of linear dependency of a given dependent variable on an specific independent variable, \( R \) and \( R^2 \) yielded the appropriate information. \( R^2 \) is the proportion of variance of the dependent variable explained or accounted
for by the independent variable. While $R^2$ is preferred because of its straightforward interpretation, $R$ was needed since it indicated the direction of the relationship.

If any of the above hypotheses were rejected, the error variance due to that variable will be reduced statistically by using analysis of covariance. Pretest scores would also be used as concomitant variables if there was found to be a significant difference between the pretest scores of the treatment groups.

Finally, to address the issue of pretest effects three separate one-way ANOVA's were preformed on the postest scores of the two control groups. If any of the following null hypotheses were rejected then the corresponding alternate hypothesis would be true and a pretest effect must be assumed.

$$H_{056}: \mu_{o,c1} = \mu_{o,c2}$$

$$H_{05P}: \mu_{P,c1} = \mu_{P,c2}$$

$$H_{05V}: \mu_{V,c1} = \mu_{V,c2}$$

$$H_{156}: \mu_{o,c1} > \mu_{o,c2}$$

$$H_{15P}: \mu_{P,c1} < \mu_{P,c2}$$

$$H_{15V}: \mu_{V,c1} > \mu_{V,c2}$$
CHAPTER FOUR
RESULTS

The purpose was to examine the effects of learning computer programming on mathematical generalization and understanding of variables. In addition, the effects of different instructional methodologies were studied. Analysis was based on both empirical and clinical data.

EMPIRICAL RESULTS

Below is a brief summary of the empirical results. Pertinent details and further information can be found in the following sections.

A. Instruments Reliability and Validity - Using the posttest scores of all the subjects, Hoyt's Reliability estimate of Cronbach's $\alpha$ was calculated. The Generalization ($R = .80$) and Programming ($R = .82$) instruments proved to be reliable but the Variable instrument had a reliability of only .67. Factor analysis was used to establish the content validity of these instruments. Results of the factor analysis supported the validity of the three instruments.
B. Comparability of research groups - To assess the comparability of the experimental groups, multivariate analysis was used to measure the group differences on the following variables: age, sex, mathematical ability, used a computer, own a computer, and length of ownership. Since Wilk's lambda was not statistically significant, it was assumed the research groups (W, E, C1, C2, and C) were comparable with respect to these variables. In addition, to assess pretest differences, three separate one-way ANOVA's were performed on the three sets of pretests scores. No evidence of differences were found and thus the experimental groups were assumed to be equivalent prior to the treatment.

C. Pretest Effect - To measure pretest effect, three separate one-way ANOVA's were performed on the posttests scores of groups C1 and C2. There were no significant differences found between the two groups' scores.

D. Treatment Effects - Subjects were randomly selected from groups C1 and C2 to form group C which was the control group used in the multivariate analysis. Using Ability as a concomitant variable, multivariate analysis of covariance was calculated and found statistically
significant \((p < .0002)\). Follow-up analysis included individual ANOVA'S using approximated mean squares for the three dependent variables and Dunn's multiple comparison test. For each dependent variable, it was determined the average mean of the groups receiving programming instruction was significantly greater than the posttest mean of the control group. The alpha levels were as follows: Generalization \((p < .1)\), Programming \((p < .005)\), and Variable \((p < .05)\).

**Instruments Reliability and Validity**

Since the three research instruments were designed by the researcher and have not been used in prior research, a discussion of the reliability and validity of these instruments is warranted.

SPSS was used to calculate the Hoyt's Reliability estimate of Cronbach's \(\alpha\). The posttest scores of all subjects were used in the analysis. The Generalization and Programming instruments proved to be reliable with \(R > .8\) (see Table 10). There may be some question of the reliability of the Variable instrument since it was shown to have a reliability of only .67.

<table>
<thead>
<tr>
<th></th>
<th>Generalization</th>
<th>Programming</th>
<th>Variable</th>
</tr>
</thead>
<tbody>
<tr>
<td>(N)</td>
<td>.80</td>
<td>.82</td>
<td>.67</td>
</tr>
</tbody>
</table>

\(N = 78\)
While reliability is a basic prerequisite for any research instrument, the most important characteristic for the instrument to possess is validity, because it is validity which measures the relationship of the data obtained to the purpose for which it was collected (Fox, 1969). Although there are four statistical procedures which can serve as an empirical basis for estimating the validity of an instrument, each of the four requires data from another instrument which is known to measure the same variable. Since such data were unavailable, it was decided to use the strongest non-statistical procedure for estimating validity; namely content validity. The procedure "argues that the instrument measures what it seeks or purports to measure because there was a rational...basis to the selection of the actual content" (Fox, 1969, p. 370).

The rational basis for each of the three research instruments was discussed previously in Chapter Three. Recall, a list of behavioral objectives was developed for each instrument and the items of the instruments were designed to measure these specific behaviors associated with mathematical generalization, programming ability, or understanding of variables. An argument for content validity can be based on evidence that these items did in fact assess achievement of the stated objectives. For this reason, factor analysis was used to establish the content validity of the Generalization, Programming, and Variable instruments. If the factors correspond to these objectives then the validity of the instruments would be supported.
Since the three instruments were administered simultaneously, the 47 items of all three were pooled and treated as a single instrument in the factor analysis. The final rotated factor matrix calculated with the SPSS program, FACTOR, was a relatively clean structure with high/low correlations and minimal variable splits. The decision to extract three factors for rotation was based upon results of the scree method. Table 11 lists the eigenvalues and percentage of the variance explained for the three rotated factors. Table 12 (p. 69) is the complete varimax rotated factor matrix after Kaiser normalization. The asterisks indicate items with loadings greater than .32.

Table 11

<table>
<thead>
<tr>
<th>Factor</th>
<th>Eigenvalue</th>
<th>Pct of Var</th>
<th>Cum Pct</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>9.43</td>
<td>20.5%</td>
<td>20.5%</td>
</tr>
<tr>
<td>II</td>
<td>3.34</td>
<td>7.3%</td>
<td>27.8%</td>
</tr>
<tr>
<td>III</td>
<td>2.81</td>
<td>6.1%</td>
<td>33.9%</td>
</tr>
</tbody>
</table>

N = 85

Item loadings greater than or equal to .32 were considered to be significant. In general, items with significant loadings grouped with respect to the three instruments. In other words, Generalization items loaded with Factor I, Programming items with Factor II, and Variable items with Factor III. Exceptions were as follows:
### Table 12
**Varimax Rotated Factor Matrix**

<table>
<thead>
<tr>
<th>Item</th>
<th>Factor 1</th>
<th>Factor 11</th>
<th>Factor II</th>
</tr>
</thead>
<tbody>
<tr>
<td>601</td>
<td>0.70*</td>
<td>0.11</td>
<td>-0.01</td>
</tr>
<tr>
<td>602</td>
<td>0.76*</td>
<td>0.05</td>
<td>0.02</td>
</tr>
<tr>
<td>603</td>
<td>0.78*</td>
<td>-0.06</td>
<td>0.34*</td>
</tr>
<tr>
<td>604</td>
<td>0.77*</td>
<td>-0.09</td>
<td>0.31</td>
</tr>
<tr>
<td>605</td>
<td>0.62*</td>
<td>0.16</td>
<td>0.03</td>
</tr>
<tr>
<td>606</td>
<td>0.64*</td>
<td>0.24</td>
<td>-0.05</td>
</tr>
<tr>
<td>607</td>
<td>0.45*</td>
<td>0.14</td>
<td>0.03</td>
</tr>
<tr>
<td>608</td>
<td>0.43*</td>
<td>0.40*</td>
<td>-0.13</td>
</tr>
<tr>
<td>609</td>
<td>0.39*</td>
<td>0.22</td>
<td>-0.04</td>
</tr>
<tr>
<td>610</td>
<td>0.39*</td>
<td>0.20</td>
<td>-0.13</td>
</tr>
<tr>
<td>611</td>
<td>0.20</td>
<td>0.01</td>
<td>-0.26</td>
</tr>
<tr>
<td>612</td>
<td>0.44*</td>
<td>0.17</td>
<td>0.08</td>
</tr>
<tr>
<td>613</td>
<td>0.36*</td>
<td>0.32*</td>
<td>-0.15</td>
</tr>
<tr>
<td>614</td>
<td>0.37*</td>
<td>0.21</td>
<td>-0.13</td>
</tr>
<tr>
<td>615</td>
<td>0.27</td>
<td>0.08</td>
<td>-0.11</td>
</tr>
<tr>
<td>616</td>
<td>0.28</td>
<td>0.23</td>
<td>0.28</td>
</tr>
<tr>
<td>617</td>
<td>0.17</td>
<td>0.29</td>
<td>0.14</td>
</tr>
<tr>
<td>618</td>
<td>0.29</td>
<td>0.13</td>
<td>0.11</td>
</tr>
<tr>
<td>619</td>
<td>0.27</td>
<td>0.20</td>
<td>0.13</td>
</tr>
<tr>
<td>620</td>
<td>0.21</td>
<td>0.24</td>
<td>0.27</td>
</tr>
<tr>
<td>P01</td>
<td>0.30</td>
<td>0.15</td>
<td>0.16</td>
</tr>
<tr>
<td>P02</td>
<td>0.08</td>
<td>0.35*</td>
<td>0.02</td>
</tr>
<tr>
<td>P03</td>
<td>0.12</td>
<td>0.51*</td>
<td>0.04</td>
</tr>
<tr>
<td>P04</td>
<td>0.15</td>
<td>0.52*</td>
<td>-0.05</td>
</tr>
<tr>
<td>P05</td>
<td>0.03</td>
<td>0.36*</td>
<td>-0.07</td>
</tr>
<tr>
<td>P06</td>
<td>0.18</td>
<td>0.65*</td>
<td>0.11</td>
</tr>
<tr>
<td>P07</td>
<td>0.00</td>
<td>0.19</td>
<td>0.22</td>
</tr>
<tr>
<td>P08</td>
<td>0.07</td>
<td>0.16</td>
<td>-0.26</td>
</tr>
<tr>
<td>P09</td>
<td>0.11</td>
<td>0.32*</td>
<td>-0.09</td>
</tr>
<tr>
<td>P10</td>
<td>0.08</td>
<td>0.11</td>
<td>-0.24</td>
</tr>
<tr>
<td>P11</td>
<td>0.10</td>
<td>0.62*</td>
<td>0.02</td>
</tr>
<tr>
<td>P12</td>
<td>0.34*</td>
<td>0.28</td>
<td>-0.17</td>
</tr>
<tr>
<td>P13</td>
<td>0.10</td>
<td>0.45*</td>
<td>0.14</td>
</tr>
<tr>
<td>P14</td>
<td>0.31</td>
<td>0.50*</td>
<td>-0.25</td>
</tr>
<tr>
<td>P15</td>
<td>0.15</td>
<td>0.64*</td>
<td>-0.07</td>
</tr>
<tr>
<td>P16</td>
<td>0.21</td>
<td>0.46*</td>
<td>0.02</td>
</tr>
<tr>
<td>P17</td>
<td>0.17</td>
<td>0.60*</td>
<td>0.12</td>
</tr>
<tr>
<td>P18</td>
<td>0.17</td>
<td>0.74*</td>
<td>0.08</td>
</tr>
<tr>
<td>V01</td>
<td>0.41*</td>
<td>0.22</td>
<td>0.17</td>
</tr>
<tr>
<td>V02</td>
<td>0.12</td>
<td>0.21</td>
<td>0.12</td>
</tr>
<tr>
<td>V03</td>
<td>0.47*</td>
<td>0.21</td>
<td>-0.02</td>
</tr>
<tr>
<td>V05</td>
<td>0.29</td>
<td>0.37*</td>
<td>0.15</td>
</tr>
<tr>
<td>V06</td>
<td>0.12</td>
<td>0.24</td>
<td>0.59*</td>
</tr>
<tr>
<td>V07</td>
<td>0.13</td>
<td>0.29</td>
<td>0.81*</td>
</tr>
<tr>
<td>V08</td>
<td>0.28</td>
<td>0.07</td>
<td>0.64*</td>
</tr>
<tr>
<td>V09</td>
<td>0.20</td>
<td>0.47*</td>
<td>0.16</td>
</tr>
</tbody>
</table>

*Statistical significance level: *p < 0.05

*Item 604 was not included in the factor analysis computations.
1. Generalization item G08 split on Factors I and II
2. Variable items, V01 and V03, loaded on Factor I not Factor III
3. Variable item V09 loaded on Factor II not Factor III

Items with loadings less than .32 and the four items which did not load with the appropriate factors should be considered non-valid measures of the dependent variables. In all probability, extraneous details which were item-specific, such as arithmetical computation, contributed to the low correlations. For example, item V04 required knowledge of the conventional order of operations which could be the reason there were no correct responses to this item.

On the other hand, the items with significant loadings can be considered valid assessments of the dependent variables. Should any of the three instruments be used in future research, precision could be improved by eliminating all items with non-significant loadings and including more items similar to the ones with significant loadings. Thus, the following items could serve as bases for refinement: items G01-G10 and G12-G13 for the Generalization instrument, items P02-P06, P09, P11, and P13-P18 for the Programming instrument, and items V06-V08 for the Variable instrument. The reliabilities of these three subsets of items can be found in Table 13. The contention that the precision of the instruments would be improved is supported by the higher reliabilities.
Table 13
Reliabilities of the Valid Subsets of The Generalization, Programming, and Variable Instruments (Cronbach's $\alpha$)

<table>
<thead>
<tr>
<th>Instrument</th>
<th>Items</th>
<th>$R$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generalization</td>
<td>G01-G10</td>
<td>.81</td>
</tr>
<tr>
<td></td>
<td>G12-G14</td>
<td></td>
</tr>
<tr>
<td>Programming</td>
<td>P02-P06, P11</td>
<td>.82</td>
</tr>
<tr>
<td></td>
<td>P13-P18</td>
<td></td>
</tr>
<tr>
<td>Variable</td>
<td>V06-V08</td>
<td>.83</td>
</tr>
</tbody>
</table>

$N = 78$

Comparability of Research Groups

It was found the research groups (W, E, C1, C2, and C) were comparable with respect to many potentially confounding variables, such as mathematical ability. To assess the comparability of the experimental groups, multivariate analysis was used to measure the group differences on the following variables: age, sex, mathematical ability, used a computer, own a computer, and length of ownership. This data, which was obtained from a background questionnaire, is summarized in Table 14.
Table 14  
Summary of Background Data by Group

<table>
<thead>
<tr>
<th>Group</th>
<th>W</th>
<th>E</th>
<th>C1</th>
<th>C2</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subjects (N)</td>
<td>26</td>
<td>29</td>
<td>28</td>
<td>23</td>
<td>29</td>
</tr>
<tr>
<td>Age</td>
<td>11.9</td>
<td>11.7</td>
<td>11.6</td>
<td>12.0</td>
<td>11.7</td>
</tr>
<tr>
<td>Sex (M/F)</td>
<td>12/14</td>
<td>14/15</td>
<td>13/15</td>
<td>11/12</td>
<td>15/14</td>
</tr>
<tr>
<td>Ability$^1$</td>
<td>54.5</td>
<td>52.6</td>
<td>53.1</td>
<td>49.7</td>
<td>52.2</td>
</tr>
<tr>
<td>(73.1) (52.7) (39.3) (47.8) (70.0)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Used Computer</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>18</td>
<td>22</td>
</tr>
<tr>
<td>Own Computer</td>
<td>10</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>11</td>
</tr>
<tr>
<td>Months</td>
<td>16.1</td>
<td>11.6</td>
<td>13.9</td>
<td>10.7</td>
<td>10.6</td>
</tr>
</tbody>
</table>

$^1$The average of students' national percentile score on the Mathematical Application subtest of the CTBS. The number in parentheses is the percentage of students who score was greater than 50. These students are considered high ability.

Specifically, two separate MANOVA's were performed; first with the means of groups W, E, C1, and C2 and then next with the means of groups W, E, and C. For the first calculation, using the two control groups, Wilk's lambda was .804 (Approx. $F = .999$). After combining the control groups, Wilk's lambda was .864 (Approx $F = .799$). Since the $F$ statistics was not statistically significant in either case, it was assumed the research groups were comparable with respect to the seven background variables.

Pretests were administered in order to identify the level of generalization, programming ability, and understanding of variables of the students prior to the treatment and to determine whether or not pretest differences would require covariance procedures in subsequent analysis. To assess pretest differences, three separate one-way
ANOVA's were performed on the three sets of pretests scores. The means of groups W, E, and C1 were used in the analysis of GEN PR and VAR PR. The means of groups W, E, and C2 were used in the analysis of PROG PR. There were no significant differences found between the three groups' scores (see Tables 15 and 16). Thus the experimental groups were assumed to be equivalent prior to the beginning of the experiment.

Table 15
Means and Standard Deviations
Of Pretest Scores by Group

<table>
<thead>
<tr>
<th></th>
<th>Gen</th>
<th></th>
<th>Prog</th>
<th></th>
<th>Var</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>N Max1</td>
<td>M (SD)</td>
<td>Max M (SD)</td>
<td>Max M (SD)</td>
<td>Max M (SD)</td>
</tr>
<tr>
<td>----------</td>
<td>--------------</td>
<td>----------</td>
<td>--------------</td>
<td>----------</td>
<td>--------------</td>
</tr>
<tr>
<td>W</td>
<td>25 29</td>
<td>13.4 (5.3)</td>
<td>29 2.6 (4.1)</td>
<td>14 2.2 (1.4)</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td>27</td>
<td>14.4 (6.5)</td>
<td>2.3 (2.1)</td>
<td>2.4 (1.6)</td>
<td></td>
</tr>
<tr>
<td>C1</td>
<td>28</td>
<td>11.6 (5.9)</td>
<td>NA*</td>
<td>2.9 (2.7)</td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td>23</td>
<td>NA</td>
<td>2.8 (2.2)</td>
<td>NA</td>
<td></td>
</tr>
</tbody>
</table>

1 Maximum score possible
2 N=26
3 N=29
4 NA means the pretest was not administered to that group.

Table 16
ANOVA Results of Mean Pretest Scores by Treatment Group

<table>
<thead>
<tr>
<th>Pretest</th>
<th>MS</th>
<th>F(2,75)</th>
<th>p &lt; .05</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEN</td>
<td>51.26</td>
<td>1.41</td>
<td></td>
</tr>
<tr>
<td>PROG</td>
<td>1.78</td>
<td>.20</td>
<td></td>
</tr>
<tr>
<td>VAR</td>
<td>2.40</td>
<td>.58</td>
<td></td>
</tr>
</tbody>
</table>

1 The scores of groups W, E, and C1 were used in the analysis of the GEN and VAR pretests. The scores of W, E, and C2 were used for the analysis of the PROG pretest.
2 For this computation, N = 75 and dfReb = 72.
3 Under the present model, the F statistic is positively biased (see Table 17).
While it is true homogeneity of variance is a basic assumption of the statistical model for analysis of variance, moderate departures from this assumption do not seriously affect the sampling distribution of the resulting $F$ statistic (Winer, 1971, p. 205). Hartley has designed a relatively simple test of homogeneity of variance. This test may be used even when the number of observations in each of the treatment groups is not constant -- if they are relatively close to being equal. Hartley's test was used on the pretest means of the three treatment groups: W, E, and C1 for the calculations of GEN PR and VAR PR and W, E, and C2 for PROG PR. It was found the hypothesis of homogeneity of variance must be rejected for PROG PR and VAR PR (see Table 17).

Table 17
Hartley's Test of Homogeneity Of Variance on Mean Pretest Scores

<table>
<thead>
<tr>
<th></th>
<th>$F_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gen Pr</td>
<td>1.50</td>
</tr>
<tr>
<td>Prog Pr</td>
<td>3.47 *</td>
</tr>
<tr>
<td>Var Pr</td>
<td>3.72 *</td>
</tr>
</tbody>
</table>

* $F(3,28) = 3.00, p < .01$

Thus it must be assumed the $F$ statistic of the ANOVA's of PROG PR and VAR PR are positive biased. In other words, the actual probability of committing an alpha error is greater than the nominal level of significance. Recall, an alpha error is committed if a true null hypothesis is rejected and mean differences are claimed to exist.
However, for the above statistical tests there was no claim of significance and the null hypotheses were not rejected. Finally, it should also be noted Winer believes there is evidence to suggest all tests of homogeneity of variance are oversensitive to departures from normality of the distributions of the basic observations.

Pretest effect

Recall, half of the control subjects took non-relevant mathematics tests in place of the three research instruments (see Tables 7 and 8, pp. 57-58). The Programming pretest was not administered to control group C1 and the Variable and Generalization pretests were not administered to group C2. To measure pretest effect, three separate one-way ANOVA's were performed on the posttests scores of groups C1 and C2. There were no significant differences found between the two groups' scores (see Tables 18 and 19).

Table 18
Means and Standard Deviations of the Control Groups' Pre- and Posttest Scores

<table>
<thead>
<tr>
<th>Group</th>
<th>C1</th>
<th>C2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N</td>
<td>M</td>
</tr>
<tr>
<td>Gen¹</td>
<td>Pre</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>(29)</td>
<td>Post</td>
</tr>
<tr>
<td>Prog²</td>
<td>Pre</td>
<td>NA</td>
</tr>
<tr>
<td></td>
<td>(29)</td>
<td>Post</td>
</tr>
<tr>
<td>Var²</td>
<td>Pre</td>
<td>28</td>
</tr>
<tr>
<td></td>
<td>(14)</td>
<td>Post</td>
</tr>
</tbody>
</table>

¹The number in parentheses is the maximum score possible for that instrument
²NA means the pretest was not administered to that group.
Table 19
ANOVA Results of Mean Posttest Scores by Control Group.

<table>
<thead>
<tr>
<th>Posttest</th>
<th>MS</th>
<th>F(1,36)</th>
<th>p &lt; .05</th>
</tr>
</thead>
<tbody>
<tr>
<td>GEN</td>
<td>10.53</td>
<td>.26</td>
<td></td>
</tr>
<tr>
<td>PROG²</td>
<td>4.45</td>
<td>.25</td>
<td></td>
</tr>
<tr>
<td>VAR</td>
<td>0.03</td>
<td>.01</td>
<td></td>
</tr>
</tbody>
</table>

¹ANOVA based on all cases with no missing data (69.9%)
²Under the present model, the F statistic is positively biased (see Table 20)

Using Hartley's test on the posttest means of the two control groups, it was found the hypothesis of homogeneity of variance must be rejected for PROG PO (see Table 20). Thus it must be assumed the F statistic of the ANOVA of PROG PO is positive biased.

Table 20
Hartley's Test of Homogeneity of Variance on the Control Group's Mean Posttest Scores

<table>
<thead>
<tr>
<th></th>
<th>Nmax</th>
<th>Fmax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gen Po</td>
<td>28</td>
<td>1.12</td>
</tr>
<tr>
<td>Prog Po</td>
<td>27</td>
<td>3.13 *</td>
</tr>
<tr>
<td>Var Po</td>
<td>23</td>
<td>2.01</td>
</tr>
</tbody>
</table>

* F(2,26) = 2.63, p < .01

Treatment Effects

Since the instruments measuring the effects of the three dependent variables were administered simultaneously, it is more appropriate to use a multivariate analysis to measure treatment effect instead of three separate analyses of variance. Subjects were randomly selected from groups C1 and C2 to form group C which was the control group used in the multivariate analysis of variance (MANOVA). Using the MANOVA
program of the SPSS package, Wilk's lambda (.680) was calculated and found to be statistically significant (approx. $F = 5.18, p < .0007$). Table 21 shows the means and standard deviations of the posttest scores of the three groups used in these calculations.

Table 21

<table>
<thead>
<tr>
<th>Group</th>
<th>N</th>
<th>Max</th>
<th>M (SD)</th>
<th>Max</th>
<th>M (SD)</th>
<th>Max</th>
<th>M (SD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>25</td>
<td>29</td>
<td>15.0 (6.5)</td>
<td>29</td>
<td>9.5 (5.0)</td>
<td>14</td>
<td>2.8 (2.3)</td>
</tr>
<tr>
<td>E</td>
<td>28</td>
<td>14.6 (6.2)</td>
<td>8.8 (5.0)</td>
<td>3.8 (2.2)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>29</td>
<td>11.9a (6.5)</td>
<td>3.3 (4.6)</td>
<td>1.8 (1.5)</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1Maximum score possible
2N = 27

While multivariate analysis is not sufficient to determine specific effects of the three dependent variables, the SPSS program, MANOVA, also performs individual analysis of variance for each dependent variable using approximated mean squares. These statistics were considered since the multivariate statistics were significant. As shown in Table 22, the F statistics were statistically significant for the Programming and Variable instruments.

Table 22

<table>
<thead>
<tr>
<th>Posttest</th>
<th>Hypoth. MS</th>
<th>Error MS</th>
<th>$F(2,75)$</th>
<th>$p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gen Po</td>
<td>66.90</td>
<td>41.58</td>
<td>1.61</td>
<td>.21</td>
</tr>
<tr>
<td>Prog Po</td>
<td>323.15</td>
<td>24.52</td>
<td>13.18</td>
<td>.00001 *</td>
</tr>
<tr>
<td>Var Po</td>
<td>24.19</td>
<td>4.29</td>
<td>5.63</td>
<td>.005 *</td>
</tr>
</tbody>
</table>
An alternate analysis which would increase the sensitivity of the F-test was also appropriate for the following reasons: the F statistic for Generalization was close to statistical significance ($p < .21$), the Generalization instrument had proved reliable, and data which could be used to reduce the error variance had been collected prior to the treatment. The decision to use covariance procedures in the analysis of treatment effect was based on examination of the correlations of the background factors with the three dependent variables. From these statistics, it was determined that Mathematical Ability should be used as a concomitant variable. More specifically, using the posttest scores of groups W, E, and C, multiple regression was used to evaluate the contribution of the potential covariates (see Table 23). Since Persistence, Motivation, and Understanding scores were obtained only for the treatment groups (Groups W and E), a separate multiple regression was performed which included these variables (see Table 24).

<table>
<thead>
<tr>
<th></th>
<th>Gen Po</th>
<th>Prog Po</th>
<th>Var Po</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sex</td>
<td>$R^2$</td>
<td>Sim R</td>
<td>F</td>
</tr>
<tr>
<td>Ability</td>
<td>.52</td>
<td>.72</td>
<td></td>
</tr>
<tr>
<td>Used</td>
<td>.52</td>
<td>.05</td>
<td></td>
</tr>
<tr>
<td>Months</td>
<td>.54</td>
<td>.15</td>
<td></td>
</tr>
<tr>
<td>Own</td>
<td>.55</td>
<td>.24</td>
<td></td>
</tr>
</tbody>
</table>

* $p < .001$
** $p < .003$
Table 24
Stepwise Multiple Regression of Posttest Scores of Groups W and E

<table>
<thead>
<tr>
<th></th>
<th>Gen Po</th>
<th></th>
<th>Prog Po</th>
<th></th>
<th>Var Po</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$R^2$</td>
<td>Sim</td>
<td>$R$</td>
<td>$F$</td>
<td>$R^2$</td>
</tr>
<tr>
<td>Sex</td>
<td>.00</td>
<td>-.07</td>
<td>5.67 *</td>
<td></td>
<td>.01</td>
</tr>
<tr>
<td>Used</td>
<td>.00</td>
<td>-.01</td>
<td></td>
<td></td>
<td>.03</td>
</tr>
<tr>
<td>Ability</td>
<td>.50</td>
<td>.70</td>
<td></td>
<td></td>
<td>.28</td>
</tr>
<tr>
<td>Months</td>
<td>.51</td>
<td>.03</td>
<td></td>
<td></td>
<td>.32</td>
</tr>
<tr>
<td>Motivate</td>
<td>.52</td>
<td>.35</td>
<td></td>
<td></td>
<td>.39</td>
</tr>
<tr>
<td>Persist</td>
<td>.52</td>
<td>.31</td>
<td></td>
<td></td>
<td>.40</td>
</tr>
<tr>
<td>Own</td>
<td>.55</td>
<td>.15</td>
<td></td>
<td></td>
<td>.44</td>
</tr>
<tr>
<td>Underst</td>
<td>.56</td>
<td>.45</td>
<td></td>
<td></td>
<td>.45</td>
</tr>
</tbody>
</table>

* $p < .001$
** $p < .003$

Thus, 55% of the variance of the GEN PO score is due to the combined effects of five variables — Sex, Ability, Used, Months, and Own. The introduction of the three additional factors, Persistence, Motivation, and Understanding, only increased the combined effect to 56%. The factors have less influence on the mean PROG PO with all eight factors only accounting for 45% of the variance. Finally, the eight factors account for 54% of the variance of VAR PO.

From the results of the multiple regression, it appears as if the contribution of the eight factors on the variance of the three dependent variable is substantial and at least some should be considered as covariates. However, closer examination of these variables and their relationships to each other rules out the use of most as covariates. Specifically, calculation of Pearson correlation coefficients between all eight factors reveal significant levels of
intercorrelation which effect the interpretability of the multiple regression results (see Table 25, p. 81). For instance, recall that Persistence, Motivation, and Understanding were rated on a five point scale by the instructors of the two treatment groups. The significant correlations between these three factors along with the negative correlation between Motivation and Group indicates a lack of reliability of this rating system. Note, the posttest scores of groups W and E (N=54) were used to calculate the correlations with Persistence, Understanding, Motivation, and the pretests. The scores of groups W, E, and C were used for all the other calculations (N=83).

The correlational results do not eliminate Mathematical Ability as a potential covariate. On the contrary, the multiple regression results indicate significant correlations between Mathematical Ability and each of the three dependent variables. Ability accounted for 52% of the variance of the GEN PO score, 47% of the PROG PO score, and 27% of the VAR PO score. Furthermore, except for Motivation, Ability is not significantly correlated with any of the other background factors. Thus Ability was used as a concomitant variable in all subsequent procedures. Using the MANOVA program of the SPSS package, Wilk's lambda (.666) was calculated and found to be statistically significant (approx. F = 4.81, p < .0002). Table 26 (see p. 82) shows the adjusted means and standard deviations of the posttest scores of the three groups used in the MANCOVA calculations.
Table 25

*Pearson Correlation Coefficients of Background Variables*

<table>
<thead>
<tr>
<th></th>
<th>Group</th>
<th>Sex</th>
<th>Age</th>
<th>Used</th>
<th>Own</th>
<th>Months</th>
<th>Persist</th>
<th>Motivate</th>
<th>Underst</th>
<th>Ability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group</td>
<td>1.00</td>
<td>.00</td>
<td>.15</td>
<td>.02</td>
<td>-.11</td>
<td>-.05</td>
<td>.29*</td>
<td>-.49*</td>
<td>.20</td>
<td>-.15</td>
</tr>
<tr>
<td>Sex</td>
<td>1.00</td>
<td>.10</td>
<td>.02</td>
<td>.03</td>
<td>.06</td>
<td>.16</td>
<td>.33*</td>
<td>.27*</td>
<td>.08</td>
<td></td>
</tr>
<tr>
<td>Age</td>
<td>1.00</td>
<td>.12</td>
<td>-.04</td>
<td>-.02</td>
<td>.11</td>
<td>-.15</td>
<td>.21</td>
<td>.01</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Used</td>
<td>1.00</td>
<td>.42*</td>
<td>.32*</td>
<td>.40*</td>
<td>.11</td>
<td>.29*</td>
<td>-.01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Own</td>
<td>1.00</td>
<td>.77*</td>
<td>.18</td>
<td>.16</td>
<td>.25</td>
<td>.11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Months</td>
<td>1.00</td>
<td>.04</td>
<td>.14</td>
<td>.11</td>
<td>.02</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Persist</td>
<td>1.00</td>
<td>.35*</td>
<td>.83*</td>
<td></td>
<td>.19</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Motivate</td>
<td>1.00</td>
<td>.47*</td>
<td>.42*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Underst</td>
<td>1.00</td>
<td>.30*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ability</td>
<td>1.00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*This indicates a statistically significant correlation. The alpha level is in parentheses.*
As shown in Table 27, the results of the three individual ANOVA's indicate the F statistics was statistically significant for each of the dependent variables.

Thus it can be assumed there were group differences with respect to generalization, programming ability, and understanding of variables. Multiple comparison test were used to determine exactly which differences between the groups' GEN PO, PROG PO, and VAR PO means were statistically significant. Since the effect of the method of programming instruction as well as the effect of programming instruction was being measured, the following null hypotheses were tested using multiple comparison procedures:

\[ H_{o2G}: \mu_{g, w} = \mu_{g, e} \]
\[ H_{o2P}: \mu_{r, w} = \mu_{r, e} \]
\( H_{0V}: M_{V,W} = M_{V,E} \)
\( H_{0P}: M_{0,AVE} = M_{0,C} \)
\( H_{03P}: M_{P,AVE} = M_{P,C} \)
\( H_{03V}: M_{V,AVE} = M_{V,C} \)

where \( M_{I,AVE} \) refers to the average mean of \( M_{i,w} \) and \( M_{i,e} \), \( I = G, P, \) or \( V \). Dunn's test was used because it is appropriate for pairwise or compound comparisons and when the treatment groups have unequal n's. Furthermore, it is the most conservative of all such procedures. Of the six hypotheses, \( H_{03G}, H_{03P}, \) and \( H_{03V} \) were rejected. In other words, for each dependent variable, the average of the mean posttest scores of the treatment groups (Groups W and E) was significantly higher than the mean score of the control group (Group C). Table 28 summarizes the post hoc calculations and results.

Table 28
Results of Dunn's Test

<table>
<thead>
<tr>
<th>Comparison</th>
<th>t</th>
<th>( t(66, p/2) )</th>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( X_{0,W} - X_{0,E} )</td>
<td>0.67</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( X_{P,W} - X_{P,E} )</td>
<td>0.63</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( X_{V,W} - X_{V,E} )</td>
<td>1.86</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( (X_{0,w}+X_{0,e})/2 - X_{0,C} )</td>
<td>2.44*</td>
<td>2.36</td>
<td>.10</td>
</tr>
<tr>
<td>( (X_{P,w}+X_{P,e})/2 - X_{P,C} )</td>
<td>5.46*</td>
<td>2.83</td>
<td>.05</td>
</tr>
<tr>
<td>( (X_{V,w}+X_{V,e})/2 - X_{V,C} )</td>
<td>3.09*</td>
<td>4.12</td>
<td>.005</td>
</tr>
</tbody>
</table>

These results indicate computer programming instruction has a significant effect on students' mathematical generalization, programming ability, and understanding of variables. The statistical results did not substantiate the claim that different instructional methods would influence these three dependent variables.
CLINICAL RESULTS

In addition to the paper and pencil instruments, further information was obtained from interviews of selected students and audiotaped recording of the instruction and student programming sessions of the two treatments. The following are descriptions and results of these case studies and observations.

Case Studies

Procedure

The case study subjects were chosen by the mathematics teacher who was instructed to select 10 students, 5 from each treatment group, with an approximately equal numbers of boys and girls. These students were chosen for their willingness and ability to communicate and varied in mathematical ability and previous programming experience. The one-on-one interviews lasted approximately 40 minutes and took place either after school or during lunch. The interviews provided an opportunity to observe students' methods of generalization in a problem solving situation and to look for differences between students of the two treatment groups.
Description of Tasks

The complete interview script as well as drawings of all manipulatives can be found in Appendix E.

1. Generalization: \((10a+5)^2 = 100a^2 + 100a + 25\)

The students were asked to look for a short cut method for multiplying a 2-digit number ending in 5 by itself. Specifically, without the aid of a calculator, the students were asked to multiply 15x15, 25x25, 35x35,...,95x95. After each multiplication problem, the students were asked if they saw any patterns between the answers that would help them find the answer to similar problems. For example, after calculating 35x35, the interviewer asked, "Do you see any patterns between these 3 problems (points to 15x15, 25x25, and 35x35) that would help you find 45x45 without multiplying?" The sequence of multiplication and looking for patterns continued until the students were unable to refine their generalization.

2. Generalization: Triangle inequality - Three lengths will form a triangle if the sum of any two is greater than the third.

Thirty nine sticks (3 each 1-13 cm) were lined up in increasing order in front of the students. These sticks were color-coded and the students were given a key to the colors. The interviewer began by making 2 triangles with sides of length 6-4-3 and 10-10-10 and then asking the students, "Can you make a triangle with any 3 lengths?" If the students said yes, the interviewer
asked them to try to make a triangle with 7-2-13. After the students seemed convinced that some lengths would not form a triangle, they were asked to find a rule for deciding if three lengths would or would not form a triangle. The students were provided with examples as needed. In second part of this task, the students were given the following five sets of lengths and asked if each would form a triangle: 100-101-100, 33-20-56, 8-7-7, 3-6-3, and 19-7-11. Finally, the students were asked to state 3 lengths that would form a triangle and 3 lengths that would not.

3. Generalization: The sum of the interior angles of a convex n-gon is $180(n-2)$.

The students were provided with drawings of various convex polygons (3 - 7 sided) which were grouped according to number of sides and with paper wedges to be used to measure the angles. The wedges ranged in size from 10° to 180° and were all multiples of ten. All the angles of the polygons were also multiples of ten.

The students were asked to look for a rule for finding the sum of the angles of any polygon. The interviewer began by demonstrating how to measure the angles of a triangle using the paper wedges and establishing the fact that all triangles had the sum of 180°. Next the students were asked if they could find the sum of the angles of a four-sided figure without measuring the angles. If the students responded, they were asked to explain their answer and then to verify it by measuring the angles of one of the 4-sided polygons. This sequence was repeated until the
students were unable to refine their method. Finally, the students were asked to find the sum of the angles of a 10-sided figure and then of a 100-sided figure.

Subjects

Table 29 is a profile of the ten case study subjects; a brief descriptions of the subjects and a summary of their interview follows. The names of the students have been changed.

Table 29

Profiles of Case Study Subjects

<table>
<thead>
<tr>
<th></th>
<th>Sex</th>
<th>Gen Pre</th>
<th>Gen Post</th>
<th>Prog Pre</th>
<th>Prog Post</th>
<th>Var Pre</th>
<th>Var Post</th>
<th>Own</th>
<th>Own Comp</th>
<th>Math</th>
<th>Reading</th>
</tr>
</thead>
<tbody>
<tr>
<td>Group W</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Don</td>
<td>M</td>
<td>12</td>
<td>22</td>
<td>6</td>
<td>18</td>
<td>4</td>
<td>9</td>
<td>Yes</td>
<td>Yes(12)</td>
<td>71</td>
<td>73</td>
</tr>
<tr>
<td>Jack</td>
<td>M</td>
<td>14</td>
<td>9</td>
<td>0</td>
<td>8</td>
<td>2</td>
<td>0</td>
<td>Yes</td>
<td>Yes(24)</td>
<td>18</td>
<td>1</td>
</tr>
<tr>
<td>Helen</td>
<td>F</td>
<td>17</td>
<td>--</td>
<td>3</td>
<td>11</td>
<td>2</td>
<td>3</td>
<td>No</td>
<td></td>
<td>65</td>
<td>69</td>
</tr>
<tr>
<td>Shawn</td>
<td>M</td>
<td>19</td>
<td>24</td>
<td>21</td>
<td>25</td>
<td>6</td>
<td>7</td>
<td>Yes</td>
<td>Yes(18)</td>
<td>78</td>
<td>99</td>
</tr>
<tr>
<td>Christy</td>
<td>F</td>
<td>8</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>No</td>
<td></td>
<td>18</td>
<td>22</td>
</tr>
</tbody>
</table>

| Group E |     |         |          |          |           |         |          |     |          |      |          |
| Matt   | M   | 16      | 16       | 5        | 12        | 3       | 5        | No  |          | 59   | 66       |
| John   | M   | 17      | 4        | 1        | 4         | 5       | 5        | No  |          | 40   | 59       |
| Tracy  | F   | 24      | 19       | 3        | 16        | 1       | 4        | No  |          | 88   | 82       |
| Janice | F   | 28      | 28       | 18       | 23        | 7       | 10       | No  |          | 96   | 92       |
| Tim    | M   | 10      | 8        | 0        | 4         | 2       | 2        | No  |          | 30   | 73       |

1Maximum score = 29
2Maximum score = 29
3Maximum score = 14
*Yes/no response depending on whether the student has a microcomputer at home. The number in parentheses is the length of ownership in months.

7CTBS mathematical application subtest national percentile score
8CTBS reading comprehension subtest national percentile score
7Did not take pretest
MATT (Group E)

Description
Matt was a quiet boy of average mathematics ability. According to his mathematics teacher, Matt worked well with his computer programming partner, John, who was also interviewed.

On both the Generalization pre- and posttest, Matt identified the correct generalization for all four situations, however, he lost many points because of arithmetical errors. While his score on the Variable posttest was not dramatically higher than the pretest, some of his posttest answers indicated an enhanced understanding of variables; as is illustrated in Figure 12. On the posttest, Matt was still unable to write an equation with two variables. Yet it appears as if he understood the relationship between the two quantities represented by M and K.

<table>
<thead>
<tr>
<th>Problem V07</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write an equation using variables for this sentence.</td>
</tr>
<tr>
<td>MIKE HAS 3 LESS BOOKS THAN KIM</td>
</tr>
<tr>
<td>(Let M stand for the number of books Matt has and K stand for the number of books Kim has)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Answers</th>
<th>Pretest</th>
<th>Postest</th>
</tr>
</thead>
<tbody>
<tr>
<td>M=3 K=6</td>
<td>M=5-3, K=5</td>
<td></td>
</tr>
</tbody>
</table>

Figure 12. Matt's pre- and posttest answers for problem 7 of the Variable instrument

During the interview, Matt spent a great deal of time on each task. However, it is difficult to assess what portion of that time was
actually spent on-task since he seldom wrote anything or used the manipulatives. When asked, Matt said he was thinking although he could not exactly explain what he was thinking about.

Response to Tasks

1. After calculating $15 \times 15$ and $25 \times 25$, he noticed these products ended with a five. This level of generalization will be referred to as an "End-digits" pattern. End-digits patterns are overgeneralization since all the information is not accounted for. Furthermore, although Matt also calculated $35 \times 35$ and $45 \times 45$ he was unable to refine his initial pattern or state any other patterns.

2. When asked to state a rule for deciding if three lengths would form a triangle, Matt said, "It won't work if one side is too small." This level of generalization will be referred to as "Spatial" because the student's rule is based on the spatial relationship of the sticks and not the relationship between the numerical lengths. While Matt's rule was an overgeneralization involving only one of the three lengths, this is not necessarily so for all spatial generalizations. Since Matt's generalization was spatial, the correctness of his answers in the second and third parts of this task depended on his ability to visualize the various lengths. Matt's lack of accurate mental representations is illustrated in two of his incorrect answers. In the second part, Matt said 3-6-3 would form a triangle while in the third part he said 63-61-86 would not. When asked to explain his answers, Matt said 3-6-3 would form a triangle because 3 is "big enough" but
63-62-86 would not form a triangle because 86 is "too big."

As previously mentioned, Matt tended to not record his work or his results unless it was suggested by the interviewer. For this task, Matt did not use the chart that had been provided to record and organize the data until the second part of this task. Thus, he had no written record of the examples and non-examples of triangles. The lack of a written record may have limited Matt's ability to generalize. Without the specific lengths as references, it seems likely that Matt would be unable to improve his spatial generalization.

3. Matt did not complete this task. He had difficulty measuring the angles and in addition, he made frequent arithmetical errors when calculating the sums.

JOHN (Group E)

Description

John was Matt's programming partner. Based on his CTBS score, John's mathematical ability would be considered average to below average. John's performance on the three posttests showed a lack of effort. This is particularly evident on the Generalization instrument where his score dropped from a 17 on the pretest to a 4 on the posttest.

Response to Tasks

1. John also noticed an End-digit pattern but it was a slight refinement of Matt's. Specifically, John observed each product ended with a 25. When asked if he saw any other patterns, John responded,
"Yes, you are always multiplying the same number together," which is simply a restatement of the task.

2. Like Matt, John's generalization was spatial, "It won't make a triangle if you have a big number and a small number." While still an overgeneralization, John's rule was a refinement of Matt's since he considered more of the information. Unlike Matt, all of John's answers in the second part were correct as were his example (20-20-20) and non-example (20-13-5) for the third part. Perhaps John could form more accurate mental representations of the lengths. An alternative explanation could be John had actually observed the numerical relationship between the lengths but was not able to verbalize this relationship.

3. After finding the sum of the angles of three, four, and five-sided polygons by measuring the angles of specific examples, John was able to state, without measuring, the sum of the angles of six-sided polygons. However, when asked to explain his answer, John said he was not sure how he got 720. It seemed as if John had lost interest at this point and so the interview was ended.

TRACY (Group E)

Description

Tracy was a good student with high mathematics ability and reading comprehension. She did well on the Generalization pretest. However, she missed the last generalization on the posttest. Her score on the
Programming posttest when compared to her pretest score indicates a substantial increase in programming ability.

During the interview, Tracy showed a lack of confidence in her generalizations. If she would encounter a counterexample to her generalization, she tended to weaken her generalization instead of looking for a new approach.

Response to Tasks

1. Tracy's first observation was an End-digit pattern. However, she undergeneralized and included the problem as well as the answer in the pattern, as illustrated in the following dialogue (Note: I stands for interviewer and T for Tracy).

   I: (Tracy multiplies 15x15) Do you see any patterns?
   T: They go by 5's.
   I: What do you mean--goes by 5's?
   T: The answer -- what you multiply it by -- goes by fives and the answer has a 5 at the end too.
   I: Ok, do you see any other patterns?
   T: This answer (points to 150) is 2 times 75.
   I: The 150?
   T: Yes.

After completing the second computation (25x25), Tracy refined her End-digit pattern and attempted to find a pattern for the rest of the product.

   I: do you see any patterns between these 2 problems? (points to 15x15 and 25x25)
   T: The twenty-fives.
   I: What about the twenty-fives?
   T: Both answers end in 25.
   I: Ok, do you see anything else?
   T: They're all even numbers?
   I: What do you mean -- they're all even numbers?
   T: The answers. The first numbers are even (points to the 2 and the 6).
   I: Ok, the 2 and the 6 are even numbers. Do you see any other patterns?
T: No.
I: Could you tell me anything about 35x35?
T: Probably something like this one (points to 625).
I: What do you mean?
T: It will end up in 25.

While her second generalization accounted for all available information, she could not use it to solve similar problems, such as 35x35. Tracy calculated 35x35 and 45x45. When asked if she saw any patterns, she resorted to the modified End-digit pattern (ends in 25).

2. When given a specific example, Tracy could explain why those lengths would or would not make a triangle. However, she was unable to state a general rule for these decisions. The following exchange was preceded by three examples.

I: Do you think you can make a triangle with lengths 6-6-12?
T: (after attempting several times) No.
I: Why not?
T: When you try, it won't make a point.
I: Why not?
T: Because they (referring to 6's) are the same size. These two together equal that (lines two 6's up with the 12).
I: Could you now state a rule for deciding when 3 lengths will or will not form a triangle?
T: If this one was a seven.
I: Could you give a rule for any 3 lengths? (Tracy says nothing) Ok, why not try 5-3-12.
T: (after trying) No.
I: Why not?
T: (fiddles with the lengths, but says nothing -- interviewer repeats question) I don't know.

Since Tracy was unable to state any sort of a rule, the interviewer decided not to pursue this problem.

3. Although Tracy claims to have had little experience measuring angles, she was very good at estimating angle measurements. This estimation skill, which was helpful in the task, is illustrated in the following dialogue.
I: (Places sheet of 4-sided polygons in front of Tracy) What do you think the sum of the angles of a 4-sided figure will be?

T: (After looking for a few seconds but not measuring any of the angles) 180? (Writes down 180 then adds 180) No, 360.

I: How did you get 360?

T: (Points to figure B1 which is a rectangle) These 2 corners are 180. Each corner is 90 and 90+90=180 and then over here these 2 are also 180. So it is 360.

After finding the sum of the angles of three and four-sided polygons, Tracy was asked to find the sum for a five sided figure without measuring. She said it would be 720° because, "180 is the sum for 3 sides and if you add those (referring to 180+180) you get 360 So 360 is the number for 4 sides. And 360+360 is 720 for 5 sides." Although her answer was incorrect, she had made a reasonable conjecture. This level of generalization will be referred to as the "Add-on" method. Tracy refined her Add-on method after she measured the angles of a five-sided polygon and calculated the sum.

T: (Measures the angles of C1, a pentagon) 540

I: Ok, do you see any patterns for finding the sum of the angles of a 6-sided figure?

T: (thinks for 45 seconds) 620.

I: How did you get that?

T: 180 added to 540. And it was added to 360.

I: So what is your rule for finding the sum for a 6-sided figure?

T: 180 plus the answer I got.

I: What answer?

T: For the 5-sided figure.

I: Fine, however you should check your addition.

T: (writes down 540+180) Oh, 720.

Tracy used the Add-on method to find the sum of the angles of a ten-sided figure. When asked if she could find another method, Tracy attempted an abstract generalization; namely 10x180 -- since the figure has ten sides. However, this method did not give the same answer as the Add-on method and Tracy was unable to correct it.
JANICE (Group E)

Description

Janice was an intelligent girl and excellent student in all subjects; as is evident by her high math ability and reading comprehension CTBS scores. Janice was very motivated and persistent and thus worked well with her programming partner who had similar characteristics.

Janice had a perfect score on the Generalization pretest and only a minor arithmetical error on posttest kept her from a second perfect score. Similarly, both her Variable pretest and posttest scores were above average. The previous summer, Janice had participated in a university-sponsored computer course. Thus prior to the treatment, her program skills were better than most of the other students. During the treatment, her instructor allowed Janice and her partner to work on their own programming projects.

Response on Tasks

1. After computing 15x15 and 25x25, Janice observed the typical End-digits pattern involving 25. She was unable to refine her generalization.

2. While Janice's generalization was spatial, it did involve all three lengths. According to Janice, three lengths would not form a triangle if "two were small compared to the third." For the third part, Janice was cautious, stating an extreme example (8-6-5) and non-example (1-13-2) of a triangle.
3. For this task, the generalization beyond the Add-on strategy requires the student to observe the sum of the angles is also a product of 180 and some number. A further refinement, which will be referred to as the "Product" strategy, is to realize that number is two less than the number of sides of the figure. After finding the sum of the angles for three and four sided figures, Janice observed and used the Product strategy to find the sum for six and ten-sided figures. However, when attempting to calculate the sum for 100-gons, she became confused and was unable to give a specific answer. The following episode is Janice's observation of the Product strategy and her attempt to use it to find the sum of the angles of a 100-gon.

I: Can you use the pattern to find the sum of the angles of a 5-sided figure will be?
J: (Multiplies 180x3) 540
I: How did you get that?
J: Well, 180x2=360 — so if you did a 5-sided figure, it would be 3 times 180 which is 540.
I: (Using multiplication, Janice correctly finds the sum for six and ten-sided figures) How would you find the sum of the angles of a 100-sided figure?
J: You count it up. Like times...I think you need to use division.
I: Why do you need division?
J: You divide the 100 into something and then you multiply that by 180.
I: Oh, so you need to find that number to multiply to 180. Do you see a pattern for finding that number?
J: Yes, 180+180. And then you just keep on timesing it...No you would add it. You would add 360+360.
I: Why would you do that?
J: That would get you the degree of the 5-sided figure.
I. (Points to 180x3) Isn't this the sum of the 5-sided figure?
J: No, 3-sided.
I: (Points to calculations for the triangles). This is for 3-sided figures. Didn't you say this was for the 5-sided figure (Again, points to 180x3)
J: Right, but I think that is wrong.
Janice's failure to find the sum for a 100-sided polygon may be partially attributed to her lack of organization. Instead of using the chart to record and organize her data, Janice did her calculations on a plain sheet of paper. Thus her answers were not labeled and while looking for a pattern by reviewing her previous work, Janice mistook 180x3 for the solution for a three-sided figure. Since Janice refuted the interviewer's attempt to correct her mistake, it is assumed other factors, besides lack of organization, inhibited her success. Recall, according to Doll (1983), "make a generalization" is the highest level of mathematical generalization. Although Janice had successfully applied her generalization and extended it to two new situations, she never specifically stated it. Without verbalization of her strategy, Janice was not able to apply it to the final task.

TIM (Group E)

Description
Tim was a discipline problem, often behaving in a manner which distracted his peers. However, during the interview, he seemed self-conscious and reserved. While Tim would be considered below average in mathematics, his CTBS reading comprehension scores were above average. Although he did poorly on all of the pretests and posttests, he was one of the few students (only 14 out of 85) who answered Variable item V05 correctly.
Response to Tasks

1. Tim had a lot of difficulty with multiplication which hindered him throughout this task. When asked to look for a pattern, Tim focused on the multiplication algorithm and not the products. "You have to remember to add all of them -- 5x5 and 5x1," he said. Eventually, Tim noticed the End-digits pattern involving 25.

2. After Tim was shown two examples, he was asked, "In general, if you were to pick any three lengths would they always form a triangle?" Tim responded affirmatively and so was asked to try to make a triangle with lengths 13-9-2. Tim spent almost five minutes trying to make a triangle with sticks of these lengths, during which he repeated the same attempts over and over. The following dialogue occurred after those attempts:

   I: Can you make a triangle?
   T: no
   I: Why not?
   T: This one (13) is too big and this (2) is too small
   I: Can you state a rule for when 3 sides will make a triangle and when they won't?
   T: They work if they all add up to 13.
   I: What do you mean? Could you give me an example.
   T: Like this one 6+4+3=13

Tim's remark about the relative size of the sticks leads one to believe he observed a spatial relationship, yet when specifically asked for a rule he stated a numerical one based on irrelevant attributes. Tim was asked to try another example using his "Rule of 13." The example (7-5-1) does not produce a triangle and so Tim fell back to his Spatial generalization.
3. There was not enough time for Tim to begin this task.

DON (Group W)

Description

Don was a good student. He had no partner but worked frequently with Jack and his partner. Don was a hard worker and did not ask many questions. However, when he did ask questions they were not "How do I do this?" but instead involved specific concepts that puzzled him.

He had a TI99/4A for about a year and seemed to work on it more frequently as the treatment progressed. He brought in several programs he had written at home. The difference between his Programming pre- and posttest scores seem indicative of his hard work at school and at home.

His improvement on the Generalization posttest is partially attributable to a reduction of arithmetical errors but also to several more correct generalizations. In addition, Don did much better on third part of the Variable posttest; namely, recognizing equations that are equivalent to a verbal statement.

Response to Tasks

1. Don's work on this first task was very interesting. In calculating 25x25 and 35x25, Don made two consistent arithmetical errors which were not detected by the interviewer. Based on his answers, Don derived a precise and functional generalization. It was not until Don tried to apply his generalization to 85x85 did the interviewer notice the errors. The following is a transcript of this episode.
I: (Explains first task and asks Don to multiply $15 \times 15$ and $25 \times 25$. Don's answers are 225 and 525 respectfully). Do you see any patterns?
D: Yes
I: Could you state a rule for finding $35 \times 35$ without multiplying?
D: 825
I: How did you figure that out?
D: Well, 25 at the end and it goes up by 3's.
I: Ok, why don't you check your answer. (Don calculates $35 \times 35$ - again making a mistake in multiplication - and gets the product of 1225) Did your rule work?
D: Almost.
I: Could you state a rule that will work?
D: Always ends in 25
I: Ok, do you see anything else?
D: It always goes up by 3's--3 then 6
I: So what is $45 \times 45$?
D: (doesn't write anything down) 2025
I: How did you find that?
D: I just added 9 to 11
I: Do you want to check? (Don multiplies $45 \times 45$) So it is 2025. Please state your rule again.
D: It goes up by 3's and always ends in 25.
I: So what is $85 \times 85$?
D: (writes in column 23, 26, 29, 32) 3225
I: How did you figure that out.
D: Just go up by 3's.
I: Could you explain your work here (points to column)
D: 23--26--going up by 3's.
I: (Points to 23) So this will represent 55?
D: Yes (points to each number) 65, 75, 85.
I: Ok, do you want to check it.
(Don multiplies $85 \times 85$=7225. Seeing his guess is wrong, he looks at his work and realizes he went up by 3 instead of multiples of 3. Don starts working on the column again and writes $32+15=47$, $47+18=65$, $65+21=86$. He is confused since his rule does not work out to the calculated answer)
I: Could you explain what you are doing?
D: I going up by 3's--15, 18, 21.
I: Where did 15, 18, 21 come from?
D: 3, 6, 9,...
(At this point, interviewer realizes errors in $25 \times 25$ and $35 \times 35$ decides to stop this situation and go on to next one)
2. In the second situation, Don only made a limited spatial generalization which focused on two of the side, "A high number and low one can't make a triangle." However, Don may have understood the problem more than he was able to express since he was very accurate when asked to modify three lengths so that a triangle could be formed.

I: Specifically, why can't 7-2-13 form a triangle?
D: The number is too small
I: Which number?
D: The 2
I: How big would it have to be so that you couldn't make a triangle?
D: 7

Yet, in the third part, his example (7-8-9) and non-example (100-1-2) of triangles were extreme and may indicate undergeneralization.

3. Don never made an accurate generalization. This may partially be due to lack of time and to the arithmetical errors he made. The one conjecture Don did make involved extraneous information, namely the fact that there are 360° in a circle.

I: Could you state a rule for finding the sum of the angles of any figure?
D: Well for a 3-sided figure it is half of a circle--360° and a 4-sided is 360°.

JACK (Group W)

Description

Jack was a hard worker and seemed anxious to please. In conversations with the instructor and classmates he seemed to have a good understanding of programming and the inherent mathematical
concepts. Yet his CTBS scores as well as most of his pretest and posttest scores were extremely low. His mathematics instructor felt Jack suffered from test anxiety.

Jack enjoyed the programming instruction and worked very well with his partner, Billy, and with Don, particularly enjoying the more challenging tasks. For the past two years, Jack has had an Apple II+ microcomputer at home. The computer belonged to his father who, according to Jack, helped Jack with at least a few of his programs.

Response to Tasks

1. Jack's response to the first task was similar to Trina's. He observed the End-digits pattern involving five, however, his pattern included the fives in the product, the problem, and the partial products. After computing 35x35, Jack modified his generalization stating the product "is going to end in 25."

2. Although shown two examples, Jack seemed unsure of the task when asked to make a triangle with lengths 7-2-13. Initially, he did nothing but then asked questions such as "Is this 13?" until he seemed comfortable.

Jack's perception of the relationship was spatial but was very restricted. He had trouble verbalizing the relationship without referring to specific examples.

I: (refers to 7-2-13) Why won't those form a triangle?
J: Because this one (the 1) is too small and one connect with the big one.
I: Now can you state a rule when 3 sides will or won't make a triangle?
J: You can put like 10-10 or 10-11...11-9 or 9-8 together.
I: So can you state a rule?
J: No.
Furthermore, from Jack's responses on the second and third parts of
this task, it appears as if he formed the misconception that the first
two lengths the interviewer mentioned were the sides of the triangle
and the third the base and the three lengths would form a triangle
only if the sides were equal (Note: dotted line indicates omission of
part of the dialogue).

I: Why won't 100-101-100 form a triangle?
J: Because you need another 101 so that they'll connect at
the top.
I: You would change one of the 100's to a 101?
J: Yes.
I: How about 33-20-56
J: You need a 56 and 2 20's
I: Ok, so why won't it make a triangle right now?
J: They aren't all the same size and they won't connect and
they'll be too long.

I: Could you tell me three lengths that will make a
triangle?
J: 6-6-5
I: Could you tell me three lengths that will not make a
triangle?
J: 8-12-19

3. After finding the sum of the angles of three and four-sided
polygons by measuring, Jack was able to state, without measuring, the
sum of the angles of five and six-sided polygons. Yet, the method he
claimed to have used would not lead to specific answers. It is
possible he used the Add-on method but was not able to verbalize it
clearly.

I: (After Jack finds the sum of a four-sided figure) Do you
see any patterns?
J: It goes down by 20's
I: In the 10's column?
J: Yes, 80 then 60
I: So for a five sided figure there will be a 40?
J: Yes
I: Do you know what the sum will be for a 5 sided figure?
J: Yes
I: What is it?
J: 540
I: Do you know what the sum will be for a 6 sided figure?
J: Yes, 720
I: Can you come up with a general rule for finding the sum of
the angles of an n sided figure.
J: It will go in 20's

HELEN (Group W)

Description

Helen was above average in both mathematical ability and reading comprehension. Helen and her partner were hard workers, who stayed on-task the entire class session. However, Helen's efforts seemed motivated by the desire to complete the task not by enjoyment or interest.

Helen was one of several students whose Programming pretest answers showed the influence of prior Logo instruction; she used the REPEAT command in several responses. For the last section of the Variable pretest, Helen chose only equations without variables; however, chose two of the correct equations on the posttest.

Response to Tasks

1. In looking for patterns, Helen would look at only part of the product or only some of the problems or occasionally the partial products. Thus her generalizations were never true. However, she was able to see many different number patterns in the set of problems. The following are some of the patterns she observed; the dotted line indicates part of the dialogue was omitted.
105

I: (She has computed 15x15, 25x25, 35x35) Do you see any patterns?
H: Oh, on each of them its like getting bigger—from 5 to 25 to 225. Like on that one (points to 15x15) that's a 25 but you have a 5 then a 25 (points to 25x25) and then a 225 (points to 35x35). You are adding on one more 2.
I: What do you think 45x45 will be?
H: 12,225

I: (She has computed 45x45 and 55x55) Do you see any patterns?
H: Well, it went from 2,025 to 3,025.
I: Do you see any other patterns
H: (Refering to partial products of 55x55) This one has 275 and the next 2750. It is just an extra zero.
I: Do you see any patterns that will help you find 65x65?
H: 4,025
I: How did you figure that out?
H: Well you're going from 2,025 to 3,025 by adding 1000.

2. Helen was very cautious during this task, asking to try an example before she made a generalization. Her initial generalization, "the sides have to be around the same lengths," was spatial and vague. Although she refined her generalization, it was still spatial and restricted. Her final generalization was "At least two of the sides have to be around the same length." Her answers in the second and third parts seemed hampered by her ability to visualize the lengths. According to Helen, 3-6-3 would form a triangle but 33-20-56 would not because "there are too many—one is too much longer—well 20 and 33 are kind of close but with the 56—it's too long for the others."

3. Helen was very successful with this task. She noticed the Product pattern quickly, immediately after calculating sum of the angles of a quadrilateral.
I: Do you see a pattern that will help you figure out the sum for a 5-sided figure?
H: 180 and 180 and one for the five sided figure. So add another 180 to it.
I: So what will it be for a 5 sided figure?
H: (Multiplies 180x3) 540

When asked to find the sum for a 10-sided polygon, Helen at first multiplied 10x180 but realized her error by studying her calculations for the five-sided polygon. At this point, she was able to verbalize her rule and find the sum for a 100-sided figure.

I: How about for a 10 sided figure?
H: (Calculates 180x10) 1,800?
I: And how did you figure that out?
H: I took 10 times 180 which is 1,800.
I: Why did you use 10?
H: Because you wanted a 10 sided figure.
I: So what is your rule for finding the sum of the angles of any sided figure?
H: (Thinks, points to 180x10) I did that one wrong.
I: Do you want to do it over?

H: (Looks over her work) Ok, 3 180's for a 5 sided figure. It would be 8x180
I: Why?
H: Well for a 5 sided figure there were 3 180's and if you subtract 5 from the 3 you get 2. So it is 2 less. So if you want a 10 sided figure it would be 2 less from 10.
I: How about for a 100 sided figure?
H: 98x180.

SHAWN (Group W)

Description
Shawn was an intelligent boy; classmates perceived Shawn as a "brain" and respected him as a computer expert. Unfortunately, Shawn lacked self-discipline. During programming instruction, he rushed through the activities in hope of playing computer games, which were
not permitted. The instructor provided Shawn with additional, more challenging activities, but he preferred to assist his peers.

Shawn enjoyed computers and owned a TI99/4A for approximately eighteen months. He frequently stayed after school to work on the Apple IIe microcomputer, often running programs he had copied from magazines.

Shawn did very well on the Generalization pretest and posttest, most points were lost because of arithmetical errors.

Response to Tasks
1. After computing 15x15 and 25x25, Shawn observed the typical End-digits pattern involving 25. He did not refine his generalization after calculating 35x35 and 45x45.
2. Shawn ended up with a convoluted rule that had as many conditions as examples he tried. For example, when asked if the lengths 8-9-11 would form a triangle, Shawn said, "Yes, because 8 is the starting number, 9 is one more and then it is okay to skip one for 11." He never stated a single general rule.
3. Shawn was quick in noticing the Add-on pattern after finding the sum of the angles of three and four-sided figures. He attempted to use the Product strategy to find the sum for a ten-sided figure but was unsuccessful, claiming the sum would be 7x180. Similarly, Shawn said the sum for 100-sided figures is 97x180.
Description

Christy was an enthusiastic girl with a cheerful personality. Her CTBS scores are indicative of below average mathematics ability and reading comprehension. During the treatment, Christy, showed enthusiasm for programming but demonstrated very little understanding. Her posttest scores showed little improvement of mathematical generalization, computer programming ability, or understanding of variables. Many of her answers on the Variable posttest appeared to be random guesses.

Response to Tasks

1. Christy was very computational-oriented, preferring the multiplication problem to looking for patterns. She did observe the End-digits pattern with 25.

2. Christy was only able to state a very vague Spatial generalization, "it won't work if you have big numbers and small numbers." For the third part, her example (12-12-12) was a correct but obvious answer while her non-example (7-18-21) was incorrect.

3. Christy did not complete this task since she had difficulty measuring the angles. She was only able to find the sum of the angles for one triangle.
Summary of the Tasks

Mathematical generalization is a cognitive process; the ability to see similarities in mathematical situations that appear externally different. For any given problem, a student is able to achieve some level of generalization. For the interview, tasks were selected that appeared to have distinct levels of generalization and required minimal mathematical ability. The following is a summary of the students’ responses to each task and a discussion of the appropriateness of each task as an assessment of mathematical generalization.

1. In the first task, the student was asked to look for short-cut method for finding the square of two-digit numbers ending in five. The most common response focussed on the end-digits. Specifically, students either stated the products always ended with a five or with a twenty-five; hence, this observation will be referred to as the End-digit generalization. The End-digits generalization is an overgeneralization since it does not account for all the information (i.e. the remaining digits) and cannot be used to find the answers to similar problems. A few students looked for patterns within each individual multiplication problem instead of between the products. Only one student stated a precise and functional generalization, however it was based on two consistent multiplication errors which were not detected by the interviewer.

This task was selected because it was decided the students would be more comfortable and gain confidence if the first task resembled
typical classroom mathematics; in this case, multiplication and number patterns. Based on the familiarity aspect, this task was the easiest of the three. However, it was probably the most difficult generalization to make since there is a big leap, in the cognitive sense, from the End-digits relationship to an abstract and functional generalization. Furthermore, some students' performance seemed hindered by the lack of arithmetical proficiency. This supports Lester's (1980) finding that computational variables are important predictors of problem solving success for elementary school subjects.

2. The triangle inequality was the relationship the students were asked to discovery in the second task. Specifically, the student, using sticks of varied lengths, was asked to find a rule for deciding if three lengths would or would not form a triangle. In order to make the most precise generalization, the student would have to focus on the numerical lengths of the sticks and look for a relationship between these numbers. A less abstract generalization would involve the spatial relationship between the sticks (i.e. comparison of the relative size of one, two, or all of the sticks). If the student's generalization was spatial, the correctness of his/her answers in the second and third parts of the task was dependent on the accuracy of the student's mental representations (i.e. his/her ability to visualize the various lengths). Student's who observed only the spatial relationship often stated extreme examples (20-20-20) and non-examples (100-1-2) of triangles.
The triangle inequality is not an extremely abstract concept. However, the organization of this task was such that the student had to focus on two distinct relationships, spatial and numerical, in order to formulate this generalization. Furthermore, there was nothing in the design of the task that pointed to the connection between these two relationships.

3. The student was asked to find a method for calculating the sum of the angles of a convex n-gon. The most common strategy involved the repeated addition of 180°. While this method, which will be referred to as the Add-on method, was effective it required the knowledge of the sum of the angles of a (n-1)-gon. In other words, to calculate the sum for a pentagon, the student would need to know 360° is the sum for a quadrilateral. Using this strategy, a student was unable to find the sum for a 100-gon. For this task, the next level of generalization requires the student to observe the sum of the angles is also a multiple of 180. A further refinement, which will be referred to as the Product method, is to realize the multiplicand is two less than the number of sides.

A student's performance was affected by his/her ability to measure angles and arithmetical competency. Several students did not finish this task due to the amount of time they needed to measure the angles of one polygon and compute the sum. For the other students, this task seemed to be an appropriate assessment of mathematical generalization, partitioning those students into two distinct groups -- students who
observed the Add-on method and those who observed the Product method.

**Analysis of the Interview Data**

The primary goal of the interviews was assessing students' generalization ability in a problem solving situation. A secondary objective was the comparison of generalization ability of students of the two treatment groups. While the interviews provided a wealth of data with respect to students' ability to generalize, there were no clear distinctions between the two groups. Because of possible confounding factors, the observed absence of qualitative differences between treatment groups should not be considered conclusive evidence of the nonexistence of treatment effect. One confounding factor may be the small sample size. The variability between case study subjects is limited by the fact that there were only five subjects from each treatment group. Secondly, as stated above, students' performances on the interview tasks were hindered by extraneous factors such as arithmetical proficiency and ability to measure angles.

Each student's overall performance on the three tasks was uniquely different from the other nine students. For example, a student may formulate a precise and functional generalization for one task and then be unable to identify any patterns for the other tasks. Because of the many irrelevant factors influencing a student's performance, a student's success on one task outweighs failures on the others. In other words, Don, Janice, and Helen should be considered as possessing
a high degree of mathematical generalization because of their individual successes.

The interview data indicates a relationship between mathematical ability and mathematical generalization. Students of high mathematical ability were more successful on the interview tasks. Furthermore, the high mathematical ability students' approaches to the tasks were such as to increase their chances of success. For instance, these students were more likely to try to account for all the information in the problem. Also, in general, the high ability students would refine not discard their generalization when confronted with a counterexample.

A final observation based on the analysis of the interview data is the importance of verbalization of the generalization. Students, such as Janice and John, who did not specifically state their strategy were unable to apply it to the most abstract situations. Recall, Janice was able to use the Product strategy to calculate the sum of the angles for six and ten-sided convex polygons but not for 100-sided polygons.

Observations

For each treatment group, the instruction and student programming sessions were audiotaped with tape recorders placed by the computers. There was one tape recorder at each table for a total of two per treatment group. At the end of the treatments, the tapes were transcribed and analyzed, with the intent of identifying factors which
support and/or clarify the statistical results.

Occasionally during the treatment, the researcher would review these tapes to assess the instructors compliance with the intentions of the two treatment. While both treatments consisted of class discussions which focussed on worksheets, the audiotapes revealed, in general, they varied extensively with respect to content and presentation and stayed within the framework of the respective treatments. The following episodes are representative of the two treatments. In the Wholistic treatment lesson, the students are presented with the problem of writing a program that will cause the computer to count by two's and the eventual outcome is the introduction of the STEP command. In the Elemental treatment lesson, the INPUT command is introduced. (Note: S1, S2,... stands for different students).

Wholistic Treatment

(The session began by reviewing a program that causes the computer to count to 1000)

10 FOR N = 1 TO 1000
20 PRINT N
30 NEXT N

I: How could we get the computer to count to 1000 by two's? 2,4,6,...,1000.
S1: Change the 1 to 2
I: OK, now what will the output be?
S2: 2,3,4,5,...
I: Right, so let's think for a minute. When you count by two's, what do you do?
S3: You want to skip one and go onto the next.
I: Yes, you need to skip one, namely 3, and go onto to 4, skip one and go onto 6, etc. There is a command to do this called STEP.
Elemental Treatment

I: You have probably seen programs where it [the computer makes a statement -- asks a question -- and you can put in a number to answer the question. We're going to be looking at such programs. (puts the following program on the board)

10 PRINT "TYPE A NUMBER"
20 INPUT N
30 PRINT N, N^2, N*2
40 END

I: What will appear on the screen when you run this program?
S1: Type a number.
I: Right, does anyone know what will appear next?
S2: A question mark.
I: Yes, the computer is waiting for you to put in a number for N. Supposed we enter 2, what will the third line be?
(no response) N equals 2, so the computer prints the value for N, N^2, N*2. That means it will print 2, 2^2, and 2*2 or 2--4--4.

At the end of the treatment, the researcher reviewed the transcripts of the tapes. Several factors were identified which may have influenced the results; descriptions of these factors follow.

1. Students' beliefs and expectations

According to Wheatley (1984), there is evidence that beliefs and expectations play a major role in mathematics problem solving. The traditional school environment often conditions students to treat each activity as a discrete task. Given a worksheet, the student's main goal is completion of the assignment regardless of correctness of answers or understanding of the material. Such a view towards mathematics is contradictory with the behavior required for generalization. The following are examples of such behavior, the girls
were in Treatment W.

EXAMPLE 1: The task was to write a program to draw the following rectangle

```
*********
*********
*********
*********
```

Julie and Jeanine seemed satisfied with their program (see below) because it drew a rectangle even though they knew it was not the same rectangle.

```
10 FOR X = 1 TO 5
20 PRINT "****"
30 NEXT X
```

EXAMPLE 2: Helen and Carla wrote a program which had the computer pick a random number (i.e. LET N = INT(RND*10)). However, as written, the random number was never used. They felt the program was correct because they did not get any error statements.

2. Levels of communication

When the conversations between students and between student and instructor of the two treatment groups are compared, the level of communications seem to differ. Whether it was between the instructor and students or between students, the discussions within the Wholistic treatment appear to be of a higher level; the students seem to be involved in more analysis and problem solving. The following are two examples from each treatment group.
Wholistic Treatment

EXAMPLE 1: Julie and Jeanine have just completed typing in the following program from their worksheet.

```
10 FOR B = 1 TO 3
20 FOR A = 1 TO 5
30 PRINT B, "JULIE"
40 NEXT A
50 NEXT B
RUN
```

Julie: Oh my gosh! Look how much we have to write! We have to write one 5 times (counts on screen) -- two 5 times, three 5 times and your name 5 x 3, 15 times!!

EXAMPLE 2: The following is a discussion between the instructor and two students about how to get the computer to count backwards from 100 to 0.

Brian: Is it FOR X = 100 TO 0?
Ins.: Why don't you try it?
(Students run the program)
Dale: What happened?
Ins.: Think about it. How do you get to 99 from 100?
Dale: Minus one. Subtract.
Ins.: Can the computer do that?
Brian: Oh, I see. How about FOR X = 100 TO 0 STEP -1

Elemental Treatment

EXAMPLE 1: The objective of this assignment is to have the computer print your initial, using only PRINT statements.

Dennis: (at the computer) How do I make the top of my D?
Scott: I don't know. Just try something.
(Very little conversation. Dennis is working on his initial while Scotty writes down the program)
Scotty: This is boring.
(The boys run the program and the D does not look correct)
Scotty: Why don't you add a few spaces in the middle line. Also don't put D's here (points to top of the initial).
EXAMPLE 2: Class discussion about a program illustrating the LET command.

```
10 LET A = 5
20 LET B = 6
30 LET C = A + B
40 PRINT C
50 END
```

Ins.: C is a combination of A and B. What is C in this program?
Class: 11
Student: Why not just put 30 LET C = 11?
Ins.: Because, the addition won't always be something you can do in your head. For example 213 + 456. The main point is the computer will do the math part of the program and give you the results.

It may be argued that the variation in communication levels was due to a difference in difficulty levels of the respective programming activities.

3. Students' mathematical ability

Observational data supported the empirical findings that mathematical generalization ability is related to a student's mathematical ability. Low ability students in Wholistic treatment tended to overgeneralize. These students had trouble distinguishing between previously learned Logo commands and the new BASIC commands. Furthermore, since in the Wholistic treatment, commands were not specifically treated but used within some application, such as drawing a rectangle, the low ability students seemed unable to discern the relevant attributes of the various commands. For example, one programming task was to create a design using at least one FOR/NEXT loop. Previously, Brian had written a program, using all PRINT
statements, that drew a face. As evident in the following dialogue, Brian is surprised by the output of his program when he modified it with a FOR/NEXT loop.

10 FOR X = 0 TO 1
20 PRINT " *** "
30 PRINT "* 0 0 *"
40 PRINT "*   *"
50 PRINT "* --- *"
60 PRINT " *** "
70 NEXT X

Ins.: Two faces, that's very nice. Is that what you expected your program to do?
Brian: No, just one.
Ins.: Why did you get two?
Brian: I don't know, we said 0 to 1.
Ins.: So what happened?
Brian: The computer won't take 0 to 1?
Ins.: No, how many numbers are from 0 to 1?
Brian: (Instructor holds up fingers as Brian counts zero, one) Oh, two.

Brian's error may evolve from either a mathematical misunderstanding or programming misunderstanding. The misconception would be mathematical if Brian did not have a correct conceptualization of zero. It is possible that Brian believes zero "is not a real number" and therefore the computer would not consider it. On the other hand, Brian may not understand the FOR/NEXT loop and believe that the number of times the computer repeats his picture is totally dependent upon the last number in the FOR statement. A possible means of diagnosing Brian's error would be to ask him the outputs of his program if line 10 was replaced by each of the following lines:
Regardless of the basis for his error, Brian may have attained a higher level of programming competency in Treatment E since each command is carefully analyzed through a series of examples. However, it is unclear if such success would have had an effect on Brian's mathematical generalization.
If the present trend continues, there will be two million computers in the U.S. public schools by 1988 (Rotenberg, 1983). This is an average of twenty computers per school. The proliferation of computers in schools is partially due to the widespread perception that computers are powerful educational tools. This perception is exemplified by Fey's (1984) remark, "Prophets of the micro-electronic age argue that computers will dramatically reshape education in the decades ahead" (p. 2). To date, there is a disturbing lack of evidence substantiating the positive effects of computer use in education. Thus, results from empirical research focusing on the educational value of computers could have important theoretical and practical implications. Not only would such results aid in the generation of theories for future research but would also provide insight for the clarification of the role of the computer in today's curriculum.

Mathematics educators, in particular, have encouraged the use of microcomputers in mathematics curricula. This enthusiasm is influenced by the belief that computer programming instruction would have a positive effect on mathematics learning. In an attempt to verify one aspect of these claims, the relationship of computer programming to students' mathematical generalization and
understanding of variables was investigated. Furthermore, the effects of two different instructional methodologies on students' mathematical generalization were studied. Based on the statistical analysis of the results, the following conclusions were drawn.

CONCLUSIONS

1. Sixth grade students can learn to program.
   The effect of computer programming instruction on the students' programming ability was statistically significant.

2. Learning computer programming enhances sixth grade students' understanding of variables.
   The effect of computer programming instruction on the students' understanding of variables was statistically significant.

3. There is preliminary evidence that programming instruction enhances sixth grade students' mathematical generalization.
   The statistical results of the effect of computer programming instruction on the students' mathematical generalization was marginally significant.

4. The researcher was unable to substantiate the claim that different instructional methods would influence the students' mathematical generalization, programming
ability, and understanding of variables.

While the adjusted mean Generalization and Programming posttest scores of Group W were higher than the mean scores of Group E, the differences were not statistically significant.

DISCUSSION

Conclusion One

In a recent survey of education professional and lay groups, over 88% of the sample felt computer programming should be included into the mathematics curriculum (NCTM, 1981, p. 12). One basis for this curricular focus is the prevalent belief that programming skills can be developed in any student at any grade level. There is little research to support this assumption. Although the finding that sixth grade students obtained a measurable level of programming ability cannot be generalized to the entire elementary school population, it is significant in its own right because of attributes common to the sixth grade population; namely, the general reasoning ability of these students.

In general, twelve year olds are in a transition from concrete operational to formal operation thought. In other words, sixth grade students are usually capable of formal reasoning but still profit from
concrete, hands-on experiences. As illustrated in the following episode, the interactive mode of the microcomputer creates a concrete environment for the otherwise abstract computer programming concepts. After a brief explanation of the LET command by their instructor, the students of Treatment E were given worksheets with the instructions to enter each program into the computer and record the output (see Appendix D, Activities 7 and 9). The following program is from one of those worksheets.

<table>
<thead>
<tr>
<th>PROGRAM</th>
<th>OUTPUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 LET Z = 10</td>
<td>10</td>
</tr>
<tr>
<td>20 PRINT Z</td>
<td>13</td>
</tr>
<tr>
<td>30 LET Z = Z+3</td>
<td>13</td>
</tr>
<tr>
<td>40 PRINT Z</td>
<td>16</td>
</tr>
<tr>
<td>50 GOTO 20</td>
<td>16</td>
</tr>
</tbody>
</table>

Marty: Oh wow! I never thought it would do that!
Peter: Oh, I see. You get Z+3, you just add 3 more each time.
Marty: You double it then add 3. (Doubling it refers to the fact that each number is printed twice)
Marty: How high will it go?
Peter: As high as you want it. You print it again and again, adding 3 each time. It will go forever.
Marty: I never thought it would do that. But now it is obvious. The computer already knows how to add, multiply, subtract, divide, percent, whatever.
Peter: I want to try something (changes line 30 to LET Z = Z+1) See, now it counts by ones. You can get it to count by anything -- even 100.
Marty: I want to see it count by 100. (They change line 30 to LET Z=Z+100 and runs the program. Next they change the program so that it counts by 1000. They notice how the columns (digits) change at different rates)
Peter: It's just like miles on a car.
Marty and Peter clearly benefited from the interactive capabilities of the microcomputer. By viewing the output, these boys, on their own, were able to analyze the program, verbally explain its function concisely, and adapt the program to other similar tasks. The clinical and statistical results support the hypothesis that the microcomputer makes programming a more concrete activity and thus sixth graders are able to learn programming.

This result also has pragmatic implications since among educators or practitioners there is no accepted level of programming competence. Inasmuch as the programming instrument proved reliable and valid, these results can serve as a measuring stick in future research and curricular development.

While the results serve to verify sixth grade students' ability to learn computer programming, there remain many unanswered questions with regard to programming instruction in elementary schools. In particular, questions such as the following, should be addressed in future research.

1. While it was shown sixth grade students were capable of learning BASIC computer programming, it is not clear if BASIC is the best programming language for this population. Which programming language is most appropriate in the elementary schools?
2. Is it true that programming skills can be developed in any student at any grade level? This study should be replicated with younger children of varying mathematical ability.

Conclusion Two

From the mathematics educator's perspective, research on students' ability to learn programming is secondary to the more crucial issues regarding the potential positive effects of computer programming instruction on mathematics learning. As Hatfield (1984) said, "We know that students can learn to write computer programs; we must now consider the more critical questions about why our mathematics students should become engaged in such tasks" (p. 2). One such issue is the effect of computer programming instruction on students' understanding of variables.

There has not been a great deal of research focusing on the relationship between programming and the understanding of variables. One such study involved fifth graders learning Logo (Milner, 1973) while the other involved college students learning BASIC (Clement, 1982). Both studies found a correspondence between programming ability and understanding of variables. The result that sixth grade students' understanding of variables improved after programming instruction, reinforces the findings of Milner and Clement and generalizes the theory to a larger population.
In today's mathematics curriculum, the concept of variable is not formally introduced until eighth grade in pre-algebra or algebra classes. The findings above, however, seem to indicate sixth grade students are able to comprehend the concept of variable when appropriately presented. In the context of programming, the concept of variable is less abstract because programming encourages exploration and manipulation of variables in what the student perceives as a natural environment.

By their very nature, algebra concepts, such as variable, are abstract. There is an abundance of mathematics education research which documents students' misconceptions and lack of understanding of these concepts (e.g. Wagner, 1981; Bright and Harvey, 1980; Clement, 1979). Wagner (1981), for example, during a clinical study of high school and middle school students observed two common misconceptions about variables:

1. Changing a variable symbol implies changing the referent.
2. The linear ordering of the alphabet corresponds to the linear ordering of the number system (p. 116).

Consequently, any aids, such as the computer, that can make abstract algebraic concepts more concrete deserve the attention of mathematics educators.
Conclusion Three

Mathematics educators have often noted the apparent connections between the cognitive processes involved in computer programming and various aspects of mathematical thinking. The result that sixth grade students' mathematical generalization improved after programming instruction provides preliminary support for this hypothesis. However, since the research theories about learning and teaching computer programming in elementary schools are still evolving, it is probably premature to draw definitive conclusions regarding the effects of computer programming instruction on mathematical generalization. Yet, the positive—although marginal—results and the relative importance of this issue justifies further investigation. Identification of possible confounding factors which may have weakened the statistical results will serve to guide the direction of future research.

1. Duration of the study.

These students were involved in approximately 20 hours of programming instruction. Recent studies suggest 50 hours of active learning as an absolute minimum to achieve any significant effects (DeCorte, 1984; Milojkovic, 1983).

2. Students' beliefs and expectations

As illustrated in Chapter Four, the students of both treatment groups tended to treat each activity as a discrete task and did not look for relationships between these activities. To encourage students to analyze
each programming task and to make and refine generalizations, a programming curriculum will need to be designed based on the philosophy of the Wholistic treatment but radically different from traditional classroom instruction.

3. Students' mathematical ability

Empirical and observational data indicate a relationship between mathematical ability and mathematical generalization and between mathematical ability and programming ability. Further research is needed to assess these relationships.

4. Instrument Validity

Based on the data from this study, the Generalization instrument proved to be a reliable and valid measure of mathematical generalization. However, the results also indicated certain items should be refined or eliminated. Analysis of the interview tasks also indicated they were appropriate assessments of generalization but required some modification. In as much as mathematical generalization is an important cognitive process which warrants further investigation, attention will need to be given to the refinement and additional testing of these instruments.
Conclusion Four

Hanson and Zweng (1984) have said, "The integration of the computer in the mathematics curriculum is at best in its infancy" (p. ix). Thus the failure to obtain statistically significant results for the hypotheses focusing on the effects of different instructional methodologies should not be construed as conclusive evidence that such differences do not exist. The hypotheses and the results obtained should be considered an initial step in the process of identifying effective uses of the computer in the mathematics curriculum. Analysis of possible confounding factors will assist the clarification of the focus of future research.

1. Levels of communication

Recall, observational data indicated a difference in the communication levels of the two treatment groups. Further research into communication levels' relationship to programming ability, mathematical generalization, and understanding of variables is warranted.

2. Effectiveness of student pairs

In the Wholistic treatment group, it appeared as if communication was better between students at different computers than between partners. The pairs of students often worked as individuals, one at the computer and the other proof-reading and writing the program onto the
worksheet. However, if a student at one computer would write a successful and/or interesting program then curiosity of others would initiate a conversation.

3. Teacher effect

It is reasonable to assume a teacher effect since the instructor of the Elemental treatment group was the classroom mathematics teacher and benefitted from familiarity with the students although it is not possible to determine the significance of this effect. An effort should be made to control teacher effect in future research.

4. Contamination of Treatment Groups

The physical arrangement of the classroom effectively separated the students of Groups W and E during the instructional period, even while they worked on the computers. Furthermore, throughout the study, the students of each treatment group were requested to not share ideas with students of the other treatment group. Although these precautions were taken it is unrealistic to believe there was no contamination of the two treatment groups. A replication of this study were the subjects are randomly assigned to treatment groups but the two groups are completely segregated is recommended.
IMPLICATIONS FOR FUTURE RESEARCH

While the results provided substantial support for the hypotheses, it was clear from the data that certain modifications with regards to instrumentation and concept definition are required prior to future research. Results indicated portions of each instrument proved to be a valid and reliable measure of the dependent variables and could be used as a basis of revision to improve the precision of the instrument. The data also provide insight into student's mathematical generalization and understanding of variables which can be used to refine the definitions of these two constructs.

Two lines of inquiry for future research are clearly suggested by the results. The first would involve replication. Specifically, the study should be repeated with a similar population but with factors found to be confounding removed. These factors include teacher effect and treatment contamination. As previously mentioned, prior to a replication study, treatment and instruments would need to be refined. In addition, the study should also be repeated with various modifications; namely, different populations should be studied and different programming languages should be used. In particular, it is recommended instruction with the programming language Logo be explored.

The second line of inquiry should address a set of additional factors which were observed in this study. These factors, which include levels of communications and student pairing, appear to have a substantial effect on programming ability and mathematical generalization.
REFERENCES


Burger, W. Personal communication, January 30, 1984.


Davis, R. B. Personal communication, November 16, 1982.


-----. *One point of view: Let kids write programs*. *Arithmetic Teacher*, 1983, 30(6), 2,56.

-----. *Definition of mathematical maturity*. Unpublished manuscript, University of Minnesota, 1968.


Troutner, J. Personal communication, April 18, 1984.


APPENDIX A
INSTRUMENTS WITH ANSWER KEYS

Generalization instrument - p. 141
Programming instrument - p. 147
Understanding of Variables instrument - p. 150
Background information questionnaire - p. 151
DIRECTIONS

1. Each set of questions will have some information in a box. Study it very carefully and look for patterns. Use this information to help you answer the questions.

2. Do NOT write on the test.

3. Put ALL your answers on the answer sheet.

4. If you need to work something out use the back of your answer sheet. For example, you may want to make a list or multiply two numbers.

5. You will NOT be getting this test back because you will be taking a similar test in about 2 months.

SAMPLE PROBLEM

\[
\begin{align*}
3! &= 3 \times 2 \times 1 = 6 \\
6! &= 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720
\end{align*}
\]

1. Solve
   a. 4! = _____
   b. 10! = _____

2. How many zeros are at the end of 100!? _____
EXAMPLE: How many different pairs can you make from these two sets \((0,1,2)\) and \((a,b,c)\)?

SOLUTION: List the pairs and count.

\[
egin{align*}
0,a & \quad 0,b & \quad 0,c \\
1,a & \quad 1,b & \quad 1,c \\
2,a & \quad 2,b & \quad 2,c
\end{align*}
\]

ANSWER: 9 pairs

NOTE: The 3 elements of the first set are 0, 1, 2. The 3 elements of the second set are a, b, c.

1. How many different pairs can you make from these two sets \((0,1,2)\) and \((d,e,f)\)?
2. How many different pairs can you make from these two sets \((x,y,z)\) and \((\theta,\#,*)\)?
3. How many different pairs can you make from these sets \((1,2,3,4)\) and \((a,b,c,d)\)?
4. How many different pairs can you make from these sets \((w,x,y,z)\) and \((q,e,r,t)\)?
5. How many different pairs can you make from \((0,1,2,3,4)\) and \((a,b,c)\)?
6. How many different pairs can you make from a set \(A\) which has 6 elements and a set \(B\) which has 3 elements?
7. How many different pairs can you make from a set \(A\) which has 13 elements and a set \(B\) which has 0 elements?
8. How many different outfits can you make if you have 8 shirts and 9 pants?
9. Think about how you solved problems 1-8. Below are some ways for solving problems of this type. Some of these methods do not work. Some work but take awhile.
   Pick the method that you feel is the quickest and most general method that works for all these problems.
   If you pick D, write your answer on the line on the answer sheet.
   A. Multiply the number of elements in the first set by the number of elements in the second set.
   B. List all of the pairs and count them.
   C. If each set has 3 elements then there are 9 pairs. If each has 4 elements then there are 16 pairs. If each has 5 elements then there are 25 pairs, and so on.
   D. None of the above. The way to find all the pairs is to...
EXAMPLES:  
\[ \begin{align*} 
3^* &= 9 \\
5^* &= 25. 
\end{align*} \]

10. \(4^* = ?\)
11. \(13^* = ?\)
12. \(8^* = ?\)
13. \(100^* = ?\)

14. Think about how you solved the problems 10-13. Below are some ways for solving problems of this type. Some do not work. Pick the method that you feel is the quickest and most general method that works for all these problems. If you pick F, write your answer on the line on the answer sheet. 

\textbf{Note} - \(N\) stands for any number you want.

A. The pattern is 4, 16, 25, 36, etc.
B. \(N^* = N + N + N\)
C. \(N^*\) means to multiply the number \(N\) by itself.
D. \(N^* = N \times N\)
E. \(3^* = 3 \times 3, \ 5^* = 5 \times 5, \ \text{and so} \ 8^* = 8 \times 8.\)
F. None of the above. The way to find \(N^*\) is ...
Let S be a set of numbers. 
3, 13, 23 are some of the elements of S. 
In other words, S = (3, 13, 23, ..., ..., ..., ...) 
The order of the elements does *not* matter!! 
We do not know the other numbers or even how many other numbers there are!!

15. Below are some possible rules for deciding if a number is an element of S. 
Pick ONE phrase that correct completes the rule. There are several correct answers. 
If you pick E, write your answer on the line on the answer sheet. 
"A NUMBER IS AN ELEMENT OF THE SET S IF ..."
A. it is a prime number. 
B. its last digit is a 3. 
C. you can write it as 2 times some number plus 3 (For example, 13 = 2x5 + 3). 
D. if you can write it as 4 times some number minus 1 (For example, 23 = 4x6 - 1) 
E. There is a different rule. The rule is...

16. It so happens that 21 is also an element of S. 
Does the rule you picked still work? (write yes or no) 

17. If you answered NO, write the letter of a rule that will work for all 4 numbers, 
S = (3, 13, 23, 21, ..., ..., ..., ...) 

18. Using your new rule, list all the numbers below that are elements of S.

1 5 7 9 16 25 103 113 2000 2009
Let $T$ be a set of figures.
Below are some of the elements of $T$.

We do not know the other figures or even how many other figures there are!!

19. Below are some possible rules for deciding if a figure is an element of $T$.
Pick ONE phrase that correct completes the rule. There are several correct answers.
If you pick E, write your answer on the line on the answer sheet.
"A FIGURE IS AN ELEMENT OF THE SET $T$ IF ..."
A. it has an even number of sides.
B. none of its sides cross.
C. it has at least 3 right angles.
D. it has exactly 4 sides.
E. There is a different rule. The rule is...

20. It so happens that this figure is also an element of $T$.

Does the rule you picked still work? (write yes or no)

21. If you answered NO, write the letter of a rule that will work for all 4 figures

22. Using your new rule, decide if each of these figures are elements of $T$. If a figure is in $T$, write the number below that figure on your answer sheet.
NAME   Key

GRADE  TEACHER

ANSWER SHEET

1. 9  2. 9  3. 16  4. 16  \{1 pt each\}
5. 15  6. 18  7. 0  8. 7d
9. \(A\) \(3\) \(pts\), \(B\) \(1\) \(pt\), \(C\) \(0\) \(pts\), \(D\) \(max\ of\ 3\) \(pts\)
10. 16  11. \(10,000\) \{1 pt each\}
12. \(164\)
13. \(10,000\)
14. \(A, B\) \(0\) \(pts\), \(C, D\) \(2\) \(pts\), \(E\) \(1\) \(pt\), \(F\) \(max\ of\ 2\) \(pts\)
15. \(A, B\) \(0\) \(pts\), \(C, D\) \(0\) \(pts\)
16. \(\{max\ of\ 3\) \(pts\\), \(C\) \(depends\ on\ answer\ of\ \#15\)
17. \(A, B, C, E\) \(1\) \(pt\), \(D\) \(0\) \(pts\)
18. \(5, 7, 9, 25, 103, 113, 2009\) \(max\ of\ 2\) \(pts\)
19. \(A, B, C, E\) \(1\) \(pt\), \(D\) \(0\) \(pts\)
20. \(\{max\ of\ 3\) \(pts\\), \(C\) \(depends\ on\ answer\ of\ \#19\)
21. 1, 4, 5 \(max\ of\ 2\) \(pts\)
I. Fill in the blank with the correct BASIC command. Select your answers from the following list of words:

END FOR/NEXT GOTO HOME (CLR, CALL CLEAR)
IF/THEN INPUT INT LET
LIST NEW PRINT REM
RND RUN STEP

1. **LIST** Command that causes the program to be displayed on the screen.
2. **HOME** Command that clears the screen (but not the memory!)
3. **INPUT** Statement that lets someone answer a question the computer asks.
4. **NEW** Command that deletes the current program from the computer's memory.
5. **REPT** Command which would be used to cause the computer to display the word "HELLO" on the screen.
6. **REPT** Command which would be used to repeat a statement seven times.

II. Evaluate each expression below. (1 pt each)

1. \( B = 3 + 2^2 - 10/2 - 1 \) \( B = \) 1
2. \( A = \text{INT}(62.56786) \) \( A = 62 \)

III. For each of the programs below, you will have to change one line in order to get the output on the right. Write the correct line in the space provided.

<table>
<thead>
<tr>
<th>PROGRAM</th>
<th>CHANGE</th>
<th>OUTPUT</th>
</tr>
</thead>
</table>
| 10 FOR \( H = 7 \) TO 12  
20 PRINT \( H \)  
30 NEXT \( H \) | line 20  
20 PRINT \( H \) | 789101112 |
| 10 FOR A = 1 TO 10  
20 PRINT A  
30 NEXT A | line 10  
10 FOR A=1 TO 10 STEP 2 | 0  
2  
4  
6  
8  
10 |
IV. Study each of the programs and answer the questions.

1. If you ran this program what would the computer print? (Write your answer in the box on the right.)

```
10 LET A = 13
20 PRINT A, A+5
```

(2 pts)

```
13 18
```

2. Number the lines of this program so that it will make the computer count to 5.

```
20 PRINT N
30 NEXT N
40 FOR N = 1 TO 5
```

(1 pt)

3. In the box below write the output of this program.

```
10 LET C = 6
20 LET D = 8
30 LET E = C+D+2
40 PRINT E
50 END
```

(1 pt)

```
16
```

4. In the box on the right, write the output of this program.

```
30 FOR K = 3 TO 6 STEP 3
40 PRINT K
50 NEXT K
```

(2 pts)

```
3 6
```
5. When you run this program it tells a person how many years it will be before he (or she) is 100 years old. Statement 30 is missing. Fill in line 30 so that the program would run correctly.

```
10 REM *** "A" IS THE AGE OF THE PERSON NOW ***
20 PRINT "HOW OLD ARE YOU NOW ",
30 INPUT A
40 C = 100 - A
50 PRINT "YOU WILL BE 100 IN ";C; " YEARS."
```

V. Write a program to do each of the following.

1. Print your name 5 times.

   
   (3 pts)

2. Allows you to type in your name and then the computer prints "HELLO (your name)".

   
   (3 pts)

3. Prints "13 x 10 = ?". If you enter the correct answer to the multiplication problem, it prints "YOU ARE CORRECT." If you enter the wrong answer it prints "SORRY, THE ANSWER IS 130."

   
   (5 pts)
UNDERSTANDING OF VARIABLE INSTRUMENT

I. DIRECTIONS: For each problem, substitute the values of the variables (the unknown) and find the value of the expression. Write your answers in the boxes. The first one is done for you.

<table>
<thead>
<tr>
<th>Values of variables</th>
<th>Expression</th>
<th>Value of expression</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. a = 2, b = 3</td>
<td>6 x a x b</td>
<td>36</td>
</tr>
<tr>
<td>2. S = 17.3</td>
<td>S</td>
<td>17.3</td>
</tr>
<tr>
<td>3. f = 100, e = 4</td>
<td>f + 3</td>
<td>103</td>
</tr>
<tr>
<td>4. g = 9</td>
<td>g x (g+1)</td>
<td>90</td>
</tr>
<tr>
<td>5. H = 1, J = 2, K = 3</td>
<td>H + J x K</td>
<td>7</td>
</tr>
<tr>
<td>6. k = m - n, n = m - 3, m =15</td>
<td>(k+n)/m</td>
<td>1</td>
</tr>
</tbody>
</table>

II. DIRECTIONS: In the box, write an equation (number sentence) using variables for each of these sentences.

7. JOHN'S SCORE IS EQUAL TO FRED'S SCORE PLUS 9.
   Let J stand for John's score and F stand for Fred's score.
   \[ J = F + 9 \]

8. MIKE HAS 3 LESS BOOKS THAN KIM.
   Let M stand for the number of books Mike has and K stand for the number of books Kim has.
   \[ M = K - 3 \]

9. THERE ARE 6 TIMES AS MANY STUDENTS AS TEACHERS AT MY SCHOOL.
   Let S stand for the number of students and T stand for the number of teachers.
   \[ S = 6 \times T \]

III. DIRECTIONS: Circle ALL of the equations that stand for the sentence below.
    TWICE A NUMBER PLUS 3 IS 9. (+2 for each correct, -1 for each wrong, 0 pt minimum)

\[ 2 \times 3 + 3 = 9 \]
\[ 3^2 + 1 = 9 \]
\[ 9 = 2 \times g + 3 \]
\[ 9 + 2 \times n + 3 \]
\[ 2 \times g = 3 + 9 \]
\[ 9 = 3 + 2 \times n \]
\[ 3 \times 3 = 9 \]
\[ 2 \times g + 3 = 9 \]
\[ 3 + 3 + 3 = 9 \]
\[ 2 \times g + 3 + 9 \]
\[ 2 \times (g + 3) = 9 \]
\[ 2 \times n + = 9 \]
BACKGROUND INFORMATION QUESTIONNAIRE

Please answer the following questions as completely as possible. Don’t worry if you cannot answer them all.

NAME_________________________ AGE_____ SEX ___MALE ___FEMALE

1. Have you ever used a computer ___YES ___NO
   If you answered NO, skip to question 4

2. Check ALL the ways you have used a computer.
   ___for games ___programming
   ___for instruction ___other

3. Check ALL the computers you have used.
   ___Apple ___VIC ___Atari
   ___IBM ___TI ___PET
   ___TRS 80 Model 1, 2 or 3 ___TRS Color Computer
   ___Timex Sinclair ___TRS Color Computer

4. Do you have a computer at home? ___YES ___NO
   If you answered NO, skip to question 7

5. What kind of computer? ______________

6. Check the phrase that best completes this statement:
   We have owned our computer for around...
   ___6 months ___1 year ___1 1/2 years
   ___2 years ___3 years ___Other_____

7. Check ALL of the following in which you have participated.
   ___computer fair ___computer math course
   ___computer camp ___computer workshop
   ___programming contest ___computer instruction in school
   ___Other____________________________

8. Check ALL computer languages that you have used.
   ___BASIC ___Logo ___PASCAL
   ___PL1 ___COBOL ___FORTRAN
   ___Assembler ___Other__________
PILOT STUDY

The pilot study can be divided into two phases -- development of the instruments and development of the programming activities for the Wholistic Treatment. The instruments and the activities were field tested and revised several times. The field test groups varied in age, mathematical background, and socio-economic status. Table 30 is a list of the various groups by grade and sex.

Table 30
Summary of Field Test Groups by Grade and Sex

<table>
<thead>
<tr>
<th>Group</th>
<th>Grade</th>
<th>N</th>
<th>Male</th>
<th>Female</th>
<th>Test Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>VAR1, GEN1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>VAR2, GEN2</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td>1</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>2</td>
<td>VAR2, GEN2</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>11</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>VAR2H, GEN2H</td>
</tr>
<tr>
<td></td>
<td>12</td>
<td></td>
<td>5</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>4</td>
<td>VAR3, GEN3</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td></td>
<td>12</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>8.6(.11)</td>
<td>120</td>
<td>86</td>
<td>34</td>
<td>VAR4, GEN4, PROG1</td>
</tr>
<tr>
<td>6.2</td>
<td>8.7(.17)</td>
<td>120</td>
<td>100</td>
<td>20</td>
<td>VAR5, GEN5, PROG1, INT1</td>
</tr>
<tr>
<td>6.3</td>
<td>8.2(.21)</td>
<td>119</td>
<td>94</td>
<td>25</td>
<td>VAR5, GEN5, PROG1</td>
</tr>
<tr>
<td>7</td>
<td>5</td>
<td>15</td>
<td>11</td>
<td>4</td>
<td>GEN6</td>
</tr>
<tr>
<td>8</td>
<td>6</td>
<td>21</td>
<td>11</td>
<td>10</td>
<td>GEN6, PROG2</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>21</td>
<td>12</td>
<td>9</td>
<td>GEN7, PROG2</td>
</tr>
<tr>
<td>10</td>
<td>3-6</td>
<td>21</td>
<td>10</td>
<td>11</td>
<td>INT2</td>
</tr>
</tbody>
</table>

¹Subgroups 6.1, 6.2, 6.3 were three separate sessions of a computer camp. The above statistic is the mean grade level and the standard deviation in parentheses.
The last column of Table 30, Test Version, indicates the version of the instruments administered to each particular group.

Programming Activities

Ideas for the specific programming activities, as well as the global philosophy for the Wholistic instructional approach, came from several sources. The TABS-Math project (Damarin, 1982) was a major influence in the development of the scope and sequence of this treatment. The approach favored by the TABS-Math project for BASIC programming instruction is to begin with structural commands, such as loops and branching, and introduce other commands as necessary for particular problems. Pilot results from this project indicate this is a successful method of programming instruction for elementary school children. Furthermore, anecdotal results seem to suggest improvement of the mathematical thinking of the students in the project.

Other ideas for the Wholistic approach were derived from observation of participants of several organized computer courses for children. During the summer of 1983, the researcher observed 359 students, ages 11-17, involved in a two-week university-operated computer camp (pilot Group 6). The set up of the camp was such that each student had approximately 4 hours of one-on-one computer time and 2 hours of classroom instruction a day. In addition, The students were able to use the computers for several hours each evening. With all of this computer time, the researcher was surprised at how little
programming these students seem to have learned. For example, each student was required to complete a project by the end of the two weeks. They were able to work individually or small groups of two or three. Many of these projects were simply long series of PRINT commands with flashy graphics added in. Since IBM-PC microcomputers were used, even the programming of flashy graphics involved little problem solving or ingenuity.

The informal atmosphere of the camp, as well as the excessive emphasis placed on the project, may have contributed to the lack of achievement by these students. However, it is hypothesized that the two distinct approaches to computer programming instruction observed by the researcher may have had a significant effect. The camp was divided up into three separate sessions (6.1, 6.2, and 6.3). In each session, the students were partitioned into 3 sections (I, II, III), each section instructed by a different set of instructors. Each section was further divided into 4 subsections, A, B, C, and D. The grouping of the students was loosely based on the students' age and his/her programming pretest score. In other words, the older students were generally placed in Section III and within each section, subsection D tended to have the highest mean pretest score and subsection A the lowest. Table 31 lists the ages and mean pretest scores of each subsection. Since the sections are not similar, it is not fair to statistically compare their posttest scores. However, even when allowance was made for the variance in age and programming ability, the researcher observed marked differences in the programming instruction
of section III compared to sections I and II. Instructors of sections I and II tended to follow the text which began with the trivial commands and used simple examples to illustrate the commands. The focus was on the commands and how they function. The researcher has since labeled this as the Elemental Approach to programming instruction.

Table 31
Mean Ages and Programming Pretest Scores of the Computer Camp Subjects by Session and Subsection

<table>
<thead>
<tr>
<th>Session</th>
<th>6.1</th>
<th>6.2</th>
<th>6.3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>N</td>
<td>Age</td>
<td>PROG</td>
</tr>
<tr>
<td>Section I</td>
<td>40</td>
<td>12.7</td>
<td>.20</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
<td>12.8</td>
<td>.00</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>12.9</td>
<td>.00</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>13.2</td>
<td>.20</td>
</tr>
<tr>
<td>D</td>
<td>10</td>
<td>11.8</td>
<td>.60</td>
</tr>
<tr>
<td>Section II</td>
<td>40</td>
<td>13.3</td>
<td>1.93</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
<td>13.9</td>
<td>.40</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>11.1</td>
<td>.30</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>14.2</td>
<td>.80</td>
</tr>
<tr>
<td>D</td>
<td>10</td>
<td>13.9</td>
<td>6.20</td>
</tr>
<tr>
<td>Section III</td>
<td>40</td>
<td>14.7</td>
<td>4.05</td>
</tr>
<tr>
<td>A</td>
<td>10</td>
<td>14.3</td>
<td>.40</td>
</tr>
<tr>
<td>B</td>
<td>9</td>
<td>14.4</td>
<td>1.33</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>14.8</td>
<td>5.30</td>
</tr>
<tr>
<td>D</td>
<td>11</td>
<td>15.1</td>
<td>8.45</td>
</tr>
</tbody>
</table>

For instance, during one class period, a section II instructor discussed INPUT command, order of operation, and the difference between using a semi-colon and comma in a print statement. Figure 13 is the
program used for this discussion.

```
10 INPUT C
20 LET F = 9/5*C +32
30 PRINT C, F
```

Figure 13. Sample program from the Computer Camp

On the other hand, the instructors of section III often began class with a problem. After discussing the problem, the programming commands need to solve the problem were introduced. This approach will be referred to as the Wholistic approach. For example, one instructor began the class by having the computer play random music and thus began a discussion on randomness and eventually the RND and INT commands. Another instructor introduced graphics commands by showing the students a program which caused a disc to move in a circular path around the screen.

A second observation site was a beginning BASIC course for young children, grades 3-6 (Group 10). The course involved 10 two hour sessions. Being so young, these children's ability to think abstractly was limited. After ten weeks, while many were able to solve problems when given specific values, they were unable to write the corresponding program using variables. The classes were informal, the children spent at least half the time working individually or in pairs on self-selected programming problems. The problems were on color-coded notecards arranged according to difficulty level and the programming commands involved. It should be noted that the students often ignored
the suggested commands and tried to solve the problems using the simple
commands such as PRINT, LET, and GOTO.

While the notecards contained many good problems and provided the
researcher with activities suggestions, it appears these students could
have benefited from more guidance. Too often, some students would
become bored (frustrated?) and spend their time on non-computer related
activities. From this observation, the researcher concluded that at
least initially some guidance is needed. Furthermore, it seemed
advisable to use older subject, capable of some abstract reasoning, to
increase the possibility of obtaining a measurable change in
generalization.

The activities developed by the researcher were piloted twice.
First with a group of ten students randomly selected from the Session
6.2 of the computer camp. These students varied in ages from 11-16.
At this point, the researcher was concerned with the appropriateness of
the topics as well as the selection of an appropriate age level. The
students met with the researcher three times. During these sessions,
the group discussed the concepts involved and developed at least
partial solutions to the programming problems. Since there was not a
computer available for these discussion, the students were encouraged
to complete the activities in between the sessions. Admittedly, none
of the students spent much additional time on the activities.

The activities were revised several times prior to the second
field test. Each time the activities were reviewed by a panel of
experts which included mathematics educators, mathematicians, computer
scientists, and elementary school teachers. The panel provided insightful comments with regard to topic suggestions, reading level, programming suggestions, and instructional methods.

One of the revised activities was piloted with Group 9 in March of 1984. The researcher led the class in a discussion of the introduction of Figures Activity 1 (see Appendix C). The students worked on this activity throughout the day on two Apple microcomputers in their classroom, under the supervision of their teacher. The next day, the researcher again discussed the worksheet with the students. Taking into account the students' unfamiliarity with the researcher, the discussion did provide the researcher with some insight into the students' thought processes and suggestions for the final revision of the activities.

**Instruments**

The following are the five research instruments developed for this study:

1. Generalization test
2. Programming test
3. Understanding of Variables test
4. Interview questions
5. Background questionnaire.

1. Generalization instrument

Initially, a series of items were collected from variety of sources. Some items were modeled after the ones used by Krutetskii (1976), Silver (1979), and Burger (1980). Specific items could not be
used since those studies used interviews and not paper/pencil measures.

In selecting items, the researcher considered problems which involved any of the following processes:

1. Finding patterns
2. Testing conjectures
3. Making hypotheses
4. Recognizing an exception to a rule
5. Generalizing a concept to a new domain
6. Detecting mathematical structure
7. Identifying what is mathematically irrelevant.
8. Stating a general formula or rule

The items chosen varied in content, format, and underlying mathematical concepts. These items were reviewed by a panel of experts. The instrument was field tested and revised seven times. Descriptive statistics for each version can be found in Table 32.

Table 32
Descriptive Statistics of the Various Versions of the Generalization Instrument

<table>
<thead>
<tr>
<th>Version</th>
<th>Group</th>
<th>N</th>
<th>R^1</th>
<th>Mean (SD)</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>6</td>
<td>.48</td>
<td>0.8 (0.3)</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>10</td>
<td>.02</td>
<td>2.1 (1.9)</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>10</td>
<td>.66</td>
<td>3.2 (2.0)</td>
<td></td>
</tr>
<tr>
<td>2H</td>
<td>4</td>
<td>8</td>
<td>.88</td>
<td>3.5 (2.6)</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>19</td>
<td>.78</td>
<td>4.7 (2.5)</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>6A</td>
<td>120</td>
<td>.85</td>
<td>13.3 (8.1)</td>
<td>32</td>
</tr>
<tr>
<td>5</td>
<td>6B</td>
<td>120</td>
<td>.88</td>
<td>18.2 (1.0)</td>
<td>44</td>
</tr>
<tr>
<td>6</td>
<td>6C</td>
<td>119</td>
<td>.84</td>
<td>16.3 (0.9)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>6</td>
<td>15</td>
<td>.87</td>
<td>9.0 (6.3)</td>
<td>26</td>
</tr>
<tr>
<td>8</td>
<td>7</td>
<td>21</td>
<td>.86</td>
<td>14.7 (4.4)</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>8</td>
<td>21</td>
<td>.83</td>
<td>21.1 (6.7)</td>
<td>32</td>
</tr>
</tbody>
</table>

^1SPSS was used to calculate the Hoyt Reliability estimate of Cronbach's α.
^2Maximum points possible for each instrument.
In Table 32, Group refers to the ten field test groups described in Table 30. Intermediate versions of the Generalization (GEN) instrument often varied radically as the researcher attempted to find the format and content that would best measure this cognitive process. Small sample sizes and relatively few items per instrument may have contributed to the low reliabilities of some of the instruments.

The first students to pilot the GEN test were fourth through ninth graders from two upper middle class professional suburb (Groups 1-3). A parallel form was also administered to eight high school students (Group 4). For these groups, the GEN test and VAR test were administered together to small groups of students. GEN1 and GEN2 were similar except for the addition of question 7c and the example for question 6. Basically, GEN2 and GEN2H varied only in reading level and use of symbolic notation. Figure 14 lists sample problems from GEN2.

6a. Finding Number Patterns
   The first number is 1.
   Any other number is 2 times the number before it.
   Write the first 5 numbers.
   -- -- -- -- --

7a. This symbol $\sum$ is called a summation sign. Here is an example of how it works.
   \[
   \sum_{n=1}^{3} 3n = 3x1 + 3x2 + 3x3 + 3x4 = 30
   \]
   Find: $\sum_{n=1}^{2} 2n = --$

Figure 14. Sample items from the second version of the Generalization instrument

Most of the students were interviewed by the researcher after completing the exam. In general the students found the items interesting but easy. From the interviews, the researcher concluded
that many of these students were involved in high degree of mathematical generalization during the exam. However test scores were low since many of the items were scored right or wrong and did not allow for arithmetical error. The format of the exam was not process-oriented and did not indicate the cognitive processes of the subject. Thus the third version of the instrument which was field tested with Group 5 was extremely different from previous versions. The test involved modified flow charts called "mathematical machines". As illustrated in Figure 15, students were given examples of specific machines, such as the ADD TWO machine, and were asked to describe the outcome of these machines, create similar machines, or identify a generalized version of the machine.

**Figure 15. Sample problems from the third version of the Generalization instrument**
While some educators felt this format was too much like programming, students did not notice the similarity. However, the students of Group 5 found the machines too cryptic and were confused by the format. When an item was reworded during the interview, a student was often able to solve it. Hence the researcher abandoned this instrument and returned to a more traditional design.

The next two versions of the instrument, which were administered at the computer camp (Group 6), again did not effectively measure the cognitive process. GEN4 and GEN5 were similar except for minor grammatical and format changes. Due to its format and notation, this instrument resembled an Algebra exam (see Figure 16).

**DIRECTIONS:** You must decide if each statement below is true for all numbers (except the complex numbers). If it is not true then give a specific example of when the statement is false. Show your work.

**EXAMPLES:**
1. \( a \times a = a + a \)
   False. \(3 \times 3 = 9\) but \(3 + 3 = 6\) and \(9 = 6\).
2. \( a \times a = a^2 \)
   True.

**PROBLEMS:**
1. \((a + b)^2 = a^2 + b^2\)
2. \(2 \times c > c\)
3. If \(b > a\) then \(1/b < 1/a\)

Figure 16. Sample items from fourth version of the Generalization instrument
The younger students seemed inhibited by the format; this may have contributed to their poor performance. Secondly, B-LN and VAR instruments were administered during one 40 minute session and most students seemed unable to concentrate for that length of time. Many did not complete the test although they turned it in before time was called. Also, patterns of answers (e.g. a student would answer all trues) seem to indicate random guessing.

The final version was based on Doll's (1983) level of mathematical thought process.

1. See a pattern
2. Utilize a pattern
3. Extend a pattern
4. Make a generalization

This version of the GEN test was administered to three separate groups (Groups 7-9). It was administered twice, revised and administered a third time. The revisions mainly involved format although one additional set of items was added. The final version of the Generalization instrument can be found in Appendix A.

The last version of the Generalization instrument proved to be reliable with $R > .8$ for all three pilot groups. The researcher found it interesting that even with the additional items excluded, the mean score of Group 9 (18.1) was higher than the mean scores of Groups 7 and 8. All three groups were from upper-middle class university towns. However, the mathematics teacher of Group 9 emphasized problem solving;
in particular the use of the computer as a problem solving tool. Since mathematical generalization is positively correlated with problem solving ability, this emphasis may explain the higher mean score of this group.

2. Programming Instrument

The original Programming test (PROG1) was computer based and was administered to Group 6. It consisted of a series of twelve programs on a disk. Each program contained at least one error, syntax or logic, which needed to be debugged. Three of the programs, of increasing difficulty, are listed in Figure 17 (p. 166).

The subject could move on to the next program after correcting the program of informing the proctor that s/he was unable to find the error. A student was stopped at the end of twenty minutes or if s/he missed two programs. Unfortunately, several external factors weakened the validity of this instrument. First, familiarity with the machine was a factor. Although the students were given some time to familiarize themselves with the IBM-PC, many had difficulty with operation during the testing period. Secondly, time was a factor; especially for the more proficient programmers. These subjects could not enter their corrections into the machine fast enough to complete the task.
Figure 17. Sample items from first version of the Programming instrument

Because of these problems, the researcher opted for a more traditional paper and pencil instrument. This instrument was field tested twice with favorable results. The descriptive statistics of these field tests can be found in Table 33. Students with no previous programming experience tended to be confused and frustrated by the wording of the questions (e.g. What is the output of this program?). However, the researcher felt this was unavoidable since understanding
certain programming terms is one objective of programming instruction. A copy of the final version of the Programming instrument can be found in Appendix A.

Table 33
Descriptive Statistics of the Various Versions of the Programming Instrument

<table>
<thead>
<tr>
<th>Version</th>
<th>Group</th>
<th>N</th>
<th>R</th>
<th>Mean (SD)</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6A</td>
<td>120</td>
<td>.67³</td>
<td>4.4 (3.7)</td>
<td>12</td>
</tr>
<tr>
<td></td>
<td>6B</td>
<td>120</td>
<td>.68</td>
<td>5.4 (0.4)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6C</td>
<td>119</td>
<td>.74</td>
<td>4.3 (0.3)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>8</td>
<td>21</td>
<td>.92³</td>
<td>14.1 (9.8)</td>
<td>31</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>21</td>
<td>.81</td>
<td>11.0 (5.8)</td>
<td></td>
</tr>
</tbody>
</table>

¹Pearson correlation coefficient for PROG PR and PROG PO.
²Maximum points possible for each instrument.
³SPSS was used to calculate the Hoyt Reliability estimate of Cronbach's α.

3. Understanding of Variables Instrument

As with the Generalization instrument, the first step was to collect a series of items from variety of sources. Some items were modeled after the ones used by Wagner (1981), Clements (1979), and Milner (1972). In selecting items, the researcher considered problems which assessed achievement of the following behavioral objectives:

1. Evaluate simple algebraic expressions
2. Translate verbal statements to symbolic equations
3. Recognize all possible equations for a verbal statement

According to Wagner (1981), students often exhibit the following misconceptions about variables:
1. Certain variables \((a,b,c)\) are always non-negative while others \((x,y,z)\) are always Real numbers.
2. A variable always has to be on the left side of the equal sign.
3. A variable stands for a specific number.
4. An operation cannot be left in an answer (e.g. A student would \(6ab\) instead of the correct answer \(2a+4b\)).
5. Different variables mean different numbers and vice versa.

The final design of the VAR instrument addressed many of these misconceptions. After being reviewed by a panel of experts, the instrument was field tested and revised six times. Descriptive statistics for each version can be found in Table 34.

| Version | Group | N  | Mean (SD) | Max  
|---------|-------|----|-----------|------|
| 1       | 1     | 6  | -1.61     | 6.7 (0.5) | 9  
| 2       | 2     | 10 | .52       | 5.7 (1.9) | 9  
| 3       | 3     | 10 | .70       | 5.6 (2.2) | 9  
| 2H      | 4     | 8  | -1.00     | 2.5 (0.6) | 3  
| 3       | 5     | 17 | .55       | 2.7 (1.4) | 13 
| 4       | 6A    | 120| .90       | 13.2 (7.1) | 24 
| 5       | 6B    | 120| .94       | 27.6 (1.3) | 46 
| 6C      | 119   |    | .92       | 27.5 (1.2) | 

\(^1\)SPSS was used to calculate the Hoyt Reliability estimate of Cronbach's \(\alpha\).

\(^2\)Maximum points possible for each instrument.

In Table 34, Group refers to the ten field test groups described in Table 30. Small sample sizes and relatively few items per instrument may have contributed to the low reliabilities of some
of the instruments.

The various versions of the VAR instrument did not vary significantly with regard to mathematic content. The major modifications involved directions, format, and reading level. Interviews with the students reassured the researcher that the items were measuring the desired objectives. However, each design seemed inappropriate. If the items involved too much notation, the younger subjects felt it resembled an Algebra exam and were inhibited. On the other hand, the first attempts were too wordy and subjects would often ignore the words and perform simple computations on the numbers in the problem. For example, several students of Group 2 answered 11 for the problem in Figure 18.

\[
\text{If } n=2r-t \text{ and } r=4, \ s=6, \text{ and } t=1 \text{ then what is the value of } n? 
\]

Figure 18. Sample item from the second version of the Variable instrument

When asked to explain his/her answer, a student would reply, "Well r is 4, s is 6, and t is 1 and 4 plus 6 plus 1 is 11."

Secondly, the researcher wanted the items which measured the students ability to evaluate simple algebraic expressions to be sequenced according to difficulty level. The pilot studies aided in determining the appropriate sequence. The final version of the Variable instrument can be found in Appendix A.
4. Interview Questions

Interviews allow the educational researcher to confirm the existence of relationships involving a cognitive process, such as generalization. According to Burger (1984), interviews are essential for theoretical model building that involve cognitive processes in children.

Burger offers several suggestions for designing questions for a research interview:

1. Select only a few children but try to get a good mixture with respect to mathematical ability and sex.
2. Do not make random selections. Have someone familiar with the children make the selection with the objective of choosing highly verbal children.
3. Pick simple problems. No one performs up to their potential during an interview situation.
4. Always use a script. Structured interviews are more reliable and easier to analyze.

The first interview questionnaire (INT1) was field tested with 8 participants of the computer camp. Table 35 (see p. 171) is a profile of these students. There was an attempt to randomly select one student from each of the for subsections of sections I and III. However, prior commitments eliminated some students while others change subsections after the selection was made.

The questions for INT1 were developed without the aid of Burgers' suggestions and the resulting data is difficult to interpret (see Figure 19, p. 171). The first three questions attempted to analyze the
Table 35
Profile of Computer Camp Interview Participants

<table>
<thead>
<tr>
<th>Student</th>
<th>Section</th>
<th>Age</th>
<th>Sex</th>
<th>PROG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Darren</td>
<td>IA</td>
<td>12</td>
<td>M</td>
<td>5</td>
</tr>
<tr>
<td>Jim</td>
<td>IC</td>
<td>12</td>
<td>M</td>
<td>6</td>
</tr>
<tr>
<td>Scott</td>
<td>IC</td>
<td>13</td>
<td>M</td>
<td>2</td>
</tr>
<tr>
<td>Lucy</td>
<td>ID</td>
<td>14</td>
<td>F</td>
<td>0</td>
</tr>
<tr>
<td>Eugene</td>
<td>IIIA</td>
<td>14</td>
<td>M</td>
<td>*</td>
</tr>
<tr>
<td>Steve</td>
<td>IIIC</td>
<td>15</td>
<td>M</td>
<td>9</td>
</tr>
<tr>
<td>Harris</td>
<td>IIIC</td>
<td>14</td>
<td>M</td>
<td>11</td>
</tr>
<tr>
<td>Rob</td>
<td>IIID</td>
<td>16</td>
<td>M</td>
<td>12</td>
</tr>
</tbody>
</table>

* Did not take posttest.

1. Can you write a program to print your name?
2. Can you write a program to print your name ten times?
3. Can you write a program that adds the numbers 1 to 10?
4. Factorial is a multiplication procedure and is symbolized as n!
   Example: 1! = 1
   2! = 2x1 = 2
   3! = 3x2x1 = 6
   10! = 10x9x8x7x6x5x4x3x2x1 = 3,628,800
   What is 6!?
   How many zeros are at the end of 1000!?
5. Three boxes, each containing billiard balls, are in a closet. One box contains only even-numbered balls, one contains only odd-numbered balls, and the third box contains a mixture of odd- and even-numbered billiard balls. All of the boxes are mislabeled. The closet is dark. You select one box and draw one billiard ball from that box. By looking at this ball and the label of that box, could you correctly label the three boxes? Why or why not?

Figure 19. Interview questions used at the computer camp
subject's programming ability and also to see if s/he could generalize the program to different situation. The remain problems were an attempt to analyze the subject's use of generalization in a mathematical content. Unfortunately, both of these problems are not simple since extensive use of problem solving heuristics is required. Many of the interview subjects were unable to work the problems or to even make a reasonable attempt.

The revised interview questionnaire was based on questions developed by Burger (1980), Silver (1979), and Krutetski (1976). These items were field tested on three girls from Group 9. These items seemed challenging but not overwhelming for the girls. The revised interview script can be found in Appendix E.

5. Background Questionnaire

Based on a review of the literature and discussions with various experts, the researcher developed a list of potential confounding variables. Data on many of these variables could be obtained from the subjects themselves. Thus a background questionnaire was developed to collect the following information about each subject:

A. Sex
B. Age
C. Previous programming experience
D. Microcomputer at home
The background questionnaire is similar to the one mailed to the participants of the aforementioned computer camp and thus was not field tested. A copy of the background questionnaire can be found in Appendix A.
APPENDIX C
PROGRAMMING ACTIVITIES - TREATMENT W
INTRODUCTION
You probably would not be very excited if you were asked to count from 1 to 1000. A computer can do simple jobs like this quickly and very accurately.

DIRECTIONS
Inside the box is a computer program. Enter this program into the computer. REMEMBER - Press [RETURN] after each line. Next run the program by typing RUN and pressing [RETURN]. Run the program at least 3 times. Write the computer output in the box on the right.

PROGRAM
10 HOME
20 FOR X = 1 TO 30
30 PRINT X
40 NEXT X

DISCUSSION QUESTIONS
1. What does this program do?
2. What does line 10 tell the computer to do?
3. How does the computer know where to start?
4. How does the computer know where to end?

5. Remove line 30 by typing the line number (in this case 30) and pressing \textbf{RETURN}, Now run the program. What happens?

6. Retype line 30. Remove line 40 and run the program. What happens? Why?

7. Change line 20 to

\begin{verbatim}
20 FOR P = 1 TO 30
\end{verbatim}

Change the X's to P's in lines 30 and 40 also. Run the program. What happens?

Change all the P's to a different letter and run this program. What happens?

8. How would you change this program so that the computer would count from 1 to 1000?
COUNTING ACTIVITY 1
TEACHING THE COMPUTER TO COUNT

DIRECTIONS: Write programs to do each of the following.

a) Count from 0 to 100.

b) Count from 2 to 20.

c) Count from 17 to 3262.

d) Count from 1 to ___ (How high can the computer count?)

e) Count from ___ to ___ (You fill in the numbers).
INTRODUCTION

So far, you have programmed the computer to count forward and backwards; starting at any number and going as high as you want. But did you know it is possible for the computer to count by two's?

DIRECTIONS:
Write a program so that the computer counts from 0 to 30. Answer the questions below.

DISCUSSION QUESTIONS

1. How do you change this program so that the computer will count from 0 to 30 by TWO'S?

2. How would you change this program so that it would count by five's?
APPLICATION:

Write programs to do each of the following:

1. Print the even numbers up to 100

2. Print the odd numbers up to 1001.

3. Print multiples of 3 up to 963.

4. Simulate a countdown by counting backwards from 100 to 0.
CONSTRUCTING SIMPLE FIGURES
ACTIVITY ONE

INTRODUCTION
Can you write a program which would draw this rectangle?

***********
***********
***********
***********
***********
***********
***********

Here is a TOUGHER QUESTION—Can you write a program with ONLY ONE PRINT command which would also draw the rectangle??

DIRECTIONS
Write programs to draw the three different rectangles. The programs can only have ONE PRINT statement. Run each of the programs to make sure it does not contain any bugs.

A. ***********

***********
***********
***********
***********
***********

PROGRAM

B. *****

*****
*****
*****
*****
*****
*****

PROGRAM
1. How are these programs similar?

2. How are these programs different?

3. How would you change program A so that it would print a rectangle that fills the entire screen?
APPLICATION

Write programs to draw each of the following figures.

1. ***** PROGRAM
   *****
   *****
   *****
   *****
   *****
   *****
   *****
   *****
   *****

2. * PROGRAM
   **
   ***
   ****
   *****
   ******

3. * PROGRAM
   **
   ***
   ****
   *****
   ******

4. Design a figure and write a program to draw that figure. Your program must have a FOR/NEXT loop!!
COUNTING ACTIVITY 2

2, 4, 6, 8, ...WHO DO WE APPRECIATE!

INTRODUCTION
In COUNTING ACTIVITY 1, you wrote programs that printed multiples of 3 up to 963 and the even numbers up to 100. Wouldn't it be nice to write one program that would print the multiples of any number you wanted?

DIRECTIONS
Write programs to print each set of multiples. Run each of the programs to make sure it does not contain any "bugs" (computer errors). Write the output of each program in the box marked OUTPUT.

A. The multiples of 5 from 5 to 40.

B. The multiples of 2 from 2 to 20.

C. The multiples of 13 less than 101.
DISCUSSION QUESTIONS

1. How are these programs similar?

2. How are these programs different?

3. How would you change program A so that it would print multiples of 19 less than 200?

EXTENSION

Write a program that prints multiples of any number.

APPLICATION

Using the program you wrote in the EXTENSION SECTION, what 3 numbers would you INPUT to have the computer print each of the following OUTPUTS.

1. multiples of 6 up to 1002?

2. multiples of 13 less than 300?

3. 13  4. 10  5. 5
   18  9   18
   23  8   31
   28  7   44
   33  6
   38  5
   4
   3
   2
   1
   0
CONSTRUCTING SIMPLE FIGURES
ACTIVITY TWO

INTRODUCTION
This activity explores punctuation in a PRINT statement and NESTED LOOPS. You may want to use this information to write your programs in ACTIVITY ONE.

DIRECTIONS
Type each program into the computer and run it. Record the output in the box.

PROGRAM ONE
10 HOME
20 FOR J=1 TO 8
30 PRINT "*"
40 NEXT J

OUTPUT

PROGRAM TWO
10 HOME
20 FOR J=1 TO 8
30 PRINT "*";
40 NEXT J

OUTPUT

PROGRAM THREE
10 HOME
20 FOR J=1 TO 8
30 PRINT "*",
40 NEXT J

OUTPUT
QUESTIONS

1. How are these 3 programs the same?

2. How are these 3 programs different?

DIRECTIONS

Type each program into the computer and run it. Record the output in the box.

PROGRAM FOUR

10 HOME
20 FOR A = 1 TO 5
30 PRINT A
40 NEXT A

PROGRAM FIVE

10 HOME
15 FOR B = 1 TO 3
20 FOR A = 1 TO 5
30 PRINT A
40 NEXT A
50 NEXT B
**PROGRAM SIX**

10 HOME
15 FOR B = 1 TO 3
20 FOR A = 1 TO 5
30 PRINT A, B
40 NEXT A
50 NEXT B

**PROGRAM SEVEN**

10 HOME
20 FOR A = 1 TO 5
30 PRINT "---"
40 NEXT A

(Put your name in the blank)

**PROGRAM EIGHT**

10 HOME
15 FOR B = 1 TO 3
20 FOR A = 1 TO 5
30 PRINT B, "---"
40 NEXT A
50 NEXT B

(Put your name in the blank)
# COMPUTER WHIZ KID CONTEST

**DIRECTIONS:** WRITE PROGRAMS THAT WILL GIVE THE FOLLOWING OUTPUTS.

<table>
<thead>
<tr>
<th>OUTPUT</th>
<th>PROGRAM</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td></td>
</tr>
<tr>
<td>10 8</td>
<td></td>
</tr>
<tr>
<td>6 4</td>
<td></td>
</tr>
<tr>
<td>2 0</td>
<td></td>
</tr>
</tbody>
</table>

```
XXXXXXXXXXXXX
XXXXXXXXXXXXX
XXXXXXXXXXXXX
*  *
*   *
*  *
*  *
*  *
```

WRITE A PROGRAM THAT LETS YOU INPUT YOUR NAME AND THEN PRINTS YOUR NAME 13 TIMES.
### COMPUTER WHIZ KID CONTEST

**DIRECTIONS:** Write outputs of the following programs.

<table>
<thead>
<tr>
<th>PROGRAM</th>
<th>OUTPUT</th>
</tr>
</thead>
</table>
| 10 FOR X = 1 TO 5 STEP 2  
20 PRINT X  
30 NEXT X | |
| 10 FOR A = 1 TO 5  
20 PRINT "WHIZ KID";  
30 NEXT A | |
| 10 FOR A = 1 TO 5  
20 FOR B = 1 TO 3  
30 PRINT "+";  
40 NEXT B  
50 NEXT A | |
| 10 LET A = 7  
20 FOR X = 1 TO A  
30 PRINT "****"  
40 NEXT X | |
| 10 PRINT "HOW OLD ARE YOU?"  
20 INPUT A  
30 LET D = 365*A  
40 PRINT YOU ARE ";D;" DAYS OLD." | |

*Note: 365*A means 365 times A.*
RANDOM RECTANGLES

INTRODUCTION
Do you know what RANDOM means?? You can program the computer to print random numbers. When you run the program you won't have ANY IDEA what numbers the computer will print!!!

DIRECTIONS
1. Type each of the following programs into the computer and run each 3 times. Record the outputs in the boxes on the right.
2. For Programs 3-8, look at the 3 outputs. Write down the LARGEST and the SMALLEST numbers the computer printed.

1. PROGRAM

10 CALL CLEAR
15 RANDOMIZE
20 FOR X = 1 TO 5
#30 LET C = RND
40 PRINT C
50 NEXT X

#For the Apple
Change
10 HOME
30 LET C = RND(1)
Remove 15 RANDOMIZE

2. PROGRAM

10 CALL CLEAR
20 PRINT "ENTER A DECIMAL NUMBER ":
30 INPUT D
40 LET I = INT(D)
50 PRINT I

OUTPUT 1  OUTPUT 2  OUTPUT 3

OUTPUT 1  OUTPUT 2  OUTPUT 3

OUTPUT 1  OUTPUT 2  OUTPUT 3
### 3. Program

<table>
<thead>
<tr>
<th>OUTPUT 1</th>
<th>OUTPUT 2</th>
<th>OUTPUT 3</th>
</tr>
</thead>
</table>

10 CALL CLEAR  
15 RANDOMIZE  
20 FOR X = 1 TO 5  
30 LET C=INT(5*RND)  
40 PRINT C  
50 NEXT X

Largest number___ Smallest number___

### 4. Program

<table>
<thead>
<tr>
<th>OUTPUT 1</th>
<th>OUTPUT 2</th>
<th>OUTPUT 3</th>
</tr>
</thead>
</table>

10 CALL CLEAR  
15 RANDOMIZE  
20 FOR X = 1 TO 5  
30 LET C=INT(5*RND)+1  
40 PRINT C  
50 NEXT X

Largest number___ Smallest number___

### 5. Program

<table>
<thead>
<tr>
<th>OUTPUT 1</th>
<th>OUTPUT 2</th>
<th>OUTPUT 3</th>
</tr>
</thead>
</table>

10 CALL CLEAR  
15 RANDOMIZE  
20 FOR X = 1 TO 5  
30 LET N=INT(17*RND(1))+1  
40 PRINT N  
50 NEXT X

Largest number___ Smallest number___

### 6. Program

<table>
<thead>
<tr>
<th>OUTPUT 1</th>
<th>OUTPUT 2</th>
<th>OUTPUT 3</th>
</tr>
</thead>
</table>

10 CALL CLEAR  
15 RANDOMIZE  
20 FOR X = 1 TO 5  
30 LET N=INT(17*RND(1))  
40 PRINT N  
50 NEXT X

Largest number___ Smallest number___
7. PROGRAM

```
10 CALL CLEAR
15 RANDOMIZE
20 FOR X = 1 TO 5
30 LET N = INT(17*RND(1)) + 1
40 PRINT N
50 NEXT X
```

Largest number ___ Smallest number ___

8. (In line 30, fill in the blank with any number.)

```
10 CALL CLEAR
15 RANDOMIZE
20 FOR X = 1 TO 5
30 LET N = INT( # * RND) + 1
40 PRINT N
50 NEXT X
```

Largest number ___ Smallest number ___

EXTENSION

Using the commands, RND and INT, can you change this program so that the computer would print random rectangles?

```
10 CALL CLEAR
20 FOR L = 1 TO 7
30 FOR W = 1 TO 13
40 PRINT "R";
50 NEXT W
60 PRINT
70 NEXT L
```
Write programs to do each of the following.

1. Draws random squares.

2. Draws random right triangles.

3. Prints random numbers between 1 and 100.

4. Prints random multiplication problems.
INTRODUCTION
What would you have to tell the computer so that it would fill the screen with this design? Can you write a program WITHOUT alot of spaces in each PRINT statement?

DIRECTIONS
Write a program to draw a simple figure.
Start your program with LINE NUMBER 50 because you will be adding lines to it.

1. Add a TAB(13) to each PRINT statement so each will look like:

   PRINT TAB(13); _____

   Run the program and record the output below.

2. Change each TAB(13) to TAB(5).
   Run the program and record the output below.
3. Change each TAB(5) to TAB(T). Add these lines-

```
10 FOR T = 1 TO 100
900 NEXT T
```

Run the program and DESCRIBE the output below.

4. Remove line 10 and line 900.
Add this line-

```
20 INPUT T
```

Run the program 3 times and input different numbers each time. Record the outputs below.

5. What number do you put in for TAB(T) so your figure is in the CENTER OF THE SCREEN? (HINT: Use the information from #3 and #4)

**EXTENSION**
Write a program to draw the figure in the introduction.
COMPUTER DESIGNS
APPLICATIONS

DIRECTIONS: WRITE PROGRAMS TO DRAW THE FIGURES BELOW.
USE THE TAB COMMAND.

1.

2.

3.

4.

5.
APPENDIX D
PROGRAMMING ACTIVITIES - TREATMENT E
Activity 2

1. Type in each program. Record the computer’s output. Explore!

A. NEW
   10 PRINT "HOW"
   20 PRINT "ARE"
   30 PRINT "YOU"
   40 END
   RUN

Output:

B. NEW
   10 PRINT "GO"
   15 PRINT "GO"
   20 PRINT "GO"
   22 PRINT "FASTER"
   25 END
   RUN

Output:

C. NEW
   5 PRINT "YOU"
   7 PRINT "ARE"
   9 PRINT "SUPER"
   11 END
   RUN

Output:

Program:

2. Write a program that will make the computer print your name three times.

Program:

3. Write a program that will make the computer print a message you choose.
Activity 3

1. NEW
   10 PRINT "HELLO"
   15 PRINT "CAN YOU SWIM?"
   20 END
   RUN

2. NEW
   10 PRINT "HELLO","FROG"
   20 PRINT "CAN YOU JUMP?"
   30 END
   RUN

3. NEW
   10 PRINT "HELLO","FROG"
   15 PRINT "CAN YOU CROAK?"
   20 END
   RUN

4. NEW
   10 PRINT "GO";
   20 PRINT "STOP"
   30 END
   RUN

5. NEW
   10 PRINT "GO",
   20 PRINT "STOP"
   30 END
   RUN

6. NEW
   6 PRINT "UP"
   4 PRINT "DOWN"
   8 END
   RUN

Add to this program by typing:

5 PRINT

Now type LIST. What happens?

Now type RUN. What happens?
Activity 4

EXPLORE

1. Type in each program. Record the computer's output. Explore!

A. NEW
10 PRINT "I DO NOT REPEAT MYSELF"
20 END
RUN

Output:

Change this program by typing:

15 GOTO 10

Run the program again.

How did you stop the program?

Output:

B. NEW
10 PRINT "X"
20 PRINT "XX"
30 PRINT "XXX"
40 END
RUN

Output:

Now make the computer print the design over and over. What statement did you add to the program?

2. Write a program that makes the computer print one of these designs over and over.

<table>
<thead>
<tr>
<th>Design</th>
<th>Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>XXXX</td>
<td>X</td>
</tr>
<tr>
<td>XXX</td>
<td>XOX</td>
</tr>
<tr>
<td>XX</td>
<td>or</td>
</tr>
<tr>
<td>X</td>
<td>XOX</td>
</tr>
<tr>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>
Activity 5

1. On the grid below, create a simple design such as a house, your initial, or a square.

2. Now write a program using PRINT statements to make the computer print your design.

3. Compare your design on the grid to your design on the computer screen. Are there any differences?

4. Can you add one statement that will make the computer print your design over and over? What statement did you add?
Type in each program. Record the computer's output. Explore!
Remember to type NEW before you type in each new program.

1. 10 LET X = 25
   20 PRINT X
   30 END

   Output:

2. 10 LET Y = 13
   20 LET C = 50
   30 PRINT
   40 PRINT C, Y, C+Y
   50 END

   Output:

   Now add these lines to the program and run it again.
   42 PRINT
   45 LET Y = 20
   46 PRINT C, Y, C+Y

   Output:

3. 5 LET M = 5
   7 LET M = M+2
   9 PRINT M
   11 END

   Output:

4. 10 LET X = 37
   20 LET Z = X+4
   30 PRINT X, Z
   40 END

   Output:

   Now change one line so that the program makes the computer print:

   29
   33

   What change did you make?
Type in each program. Record the computer's output. Explore!

1. 10 LET A = 5
   20 LET B = 6
   30 LET C = A + B
   40 PRINT C
   50 END

   Output:

2. 12 LET A = 5 + 8 + 3 - 2
   20 PRINT A
   32 END

   Output:

3. 10 LET A = 14
   20 LET B = 7
   30 PRINT A+B, A-B, A*B, A/B
   40 END

   Output:

4. 10 LET Z = 10
   20 PRINT Z
   30 LET Z = Z + 3
   40 PRINT Z
   55 END

   Add this line:
   50 GOTO 10

   Run the program to see what happens.

  Output:

What happens when you type in:

50 GOTO 30

Output:

Now try this one:

50 GOTO 20

Output:
Activity 12

Read each program and write what you think the program will make the computer do. Then check your predictions by running each program.

1. 10 LET A = 6
   20 PRINT "MY NUMBER IS", A
   30 LET B = 20
   40 LET Z = 30
   50 PRINT B, Z, Z - B
   60 END

Predicted Output:

2. 1 LET C = 4
   2 LET D = 5 + 2
   3 LET E = C + 3
   4 PRINT C, D, E
   5 GOTO 1
   6 END

Predicted Output:

3. 10 PRINT "SUPERMAN"
   20 PRINT "BIONIC WOMAN"
   30 GOTO 20
   40 PRINT
   50 END

Predicted Output:

What happens if this line is typed in:

10 PRINT "SUPERMAN",

Predicted Output:

EXTRA FOR EXPERTS

10 LET C = 1
   20 LET D = 3
   25 LET F = C + D
   30 PRINT C, D, F
   40 LET C = C + 1
   50 LET D = D + C
   60 GOTO 25
   70 END

Predicted Output:
Write a program to solve each of the following problems. Remember, writing a program includes typing it into the computer and debugging it.

1. Have the computer print the sequence of counting numbers.

   1
   2
   3
   4
   ...
   ...

Program:

2. Have the computer print the multiples of two starting at two (that is, the even numbers).

   2
   4
   6
   8
   ...
   ...

Program:
Write a program to solve each of the following problems.

1. Have the computer print the sequence of numbers divisible by 21 starting with 21.
   
   Program:
   
   21
   42
   63
   

2. Make the computer print the sequence of numbers:
   
   Program:
   
   100
   95
   90
   85
   
   What happens when the computer's output reaches zero?
Activity 15

Type in each program. Record the computer’s output. Explore!

1. 10 PRINT TAB(16); "HELLO"
   20 PRINT TAB(20); "THERE"
   30 PRINT TAB(3); "EVERYONE"
   40 END

   Output:  
   207

2. 5 PRINT TAB(25); "X"
   10 LET X = 5
   20 PRINT TAB(25); X
   30 LET X = -5
   40 PRINT TAB(25); X
   50 END

   Output:  
   207

3. 11 LET X = X + 1
   17 PRINT TAB(X); X
   30 GOTO 11
   40 END

   What happens if this line is typed in:
   11 LET X = X + 2

4. 3 LET X = 5
   5 LET X = X + 4
   10 PRINT TAB(X); "SUPERKID"
   20 GOTO 5
   30 END

   What happens if these lines are typed in:
   3 LET X = 30
   5 LET X = X - 2

5. 5 REM KING ARTHUR
   10 REM "DEFEATS"
   20 PRINT "THE BIONIC DRAGON"
   30 GOTO 5
   40 END

   Output:  
   207

EXPLORE
Activity 20

Type in each program. Record the computer’s output. Explore!

1. 10 PRINT "TYPE A NUMBER"
   20 INPUT N
   30 PRINT N, 2 * N, N * N
   40 END

2. 5 PRINT "TYPE TWO NUMBERS BETWEEN 1 AND 15."
   10 PRINT "PRESS RETURN AFTER EACH NUMBER YOU TYPE."
   15 INPUT X
   20 INPUT B
   25 PRINT TAB(X);X;"IS HERE."
   30 PRINT TAB(B);B;"IS HERE."
   35 END

   Add this line:
   33 PRINT TAB(X+B);X+B;"IS HERE."

   Output:
   What happens when both
   X and B are between 20
   and 30?
   Between 30 and 40?
   Greater than 40?

3. 10 PRINT "TYPE YOUR FIRST NUMBER."
   15 INPUT M
   20 PRINT "TYPE YOUR SECOND NUMBER."
   25 INPUT N
   30 LET S = M + N
   40 PRINT "THEIR SUM IS ";S
   50 PRINT
   60 GOTO 10
   70 END

   Output:

   Program:
   Change this program to
   find the product of any
   three numbers.
Solve each of the problems below. Write your programs on a separate sheet of paper.

1. Write a program that will allow you to input the number of hours you slept last night. Have the computer print the total number of hours and the total number of days you will sleep in a year at that rate.
   Suggested output:
   
   HOW MANY HOURS DID YOU SLEEP LAST NIGHT?
   ?
   NUMBER OF HOURS YOU WILL SLEEP IN A YEAR:
   NUMBER OF DAYS YOU WILL SLEEP IN A YEAR:

2. You are the proud new manager of the Big Dipper Ice Cream Store and you need the help of a computer. Write a program that will allow you to input the number of single scoop (32¢), double scoop (60¢), and triple scoop (88¢) cones sold during the whole day. Have the computer print out the total number of scoops of ice cream sold that day and the total amount of money paid for ice cream cones that day.
   Suggested output:
   
   HOW MANY SCOOPS OF EACH: (SINGLE, DOUBLE, TRIPLE)?
   ?
   THERE WERE ___ SCOPS SOLD TODAY.
   TOTAL AMOUNT OF MONEY PAID FOR CONES TODAY WAS ___.

3. Using INPUT statements, write a program that makes the computer print the area of any triangle, given the base and height of the triangle.
   (area = 1/2 * base * height).
1. The boxes below represent part of a computer's memory. For each line in the program, record what variable names and numbers are stored together in the computer's memory.

10 LET X = 50
20 LET Y = X/2
25 LET X = 7
30 LET Z = Y + 2
40 LET B = Y * X
45 PRINT X, Z, B, A
50 GOTO 40
60 LET A = 1
65 PRINT X, Z, B, A
70 END

2. If you were to run the program in problem 1, what would be printed out? Predicted Output:

What would happen if line 50 were deleted? Predicted Output:

3. Predict what the following program will make the computer do. Run the program to check your prediction.

5 REM M = 3
10 LET X = 13
20 LET X = 25
30 PRINT X, 2 * X, X * 3, X * M
40 LET Z = 20
50 PRINT Z * 2
60 END

Predicted Output:
Activity 23

Solve each of the problems below. Write your programs on a separate sheet of paper.

1. Have the computer print your own name over and over again, in each of the following patterns.

   A. NAME
   B. NAME
   C. NAME NAME NAME NAME NAME NAME...

   NAME
   NAME
   NAME

   D. NAME NAME NAME NAME...

   NAME
   NAME
   NAME

   E. NAME
   F. NAME

   NAME
   NAME
   NAME

2. Tell the computer to count by twos from 15 to 1 in descending order.

3. Using an INPUT statement, redo problem 2 to make the computer count by twos from your chosen number to 1 in descending order. What happens if the number you input is:
   a) an even number;
   b) zero;
   c) a decimal number?
Type in each program. See what happens. Explore!

1. 
   3 LET C = 0
   5 LET C = C + 1
   7 IF C = 40 THEN GOTO 35
   10 LET X = INT(10*RND( )) + 1
   20 PRINT X,
   30 GOTO 5
   35 END

   What do you notice about the numbers output by the computer? Is there any pattern? Run the program five times and record the lowest and highest numbers of each run.

<table>
<thead>
<tr>
<th>Run 1</th>
<th>Run 2</th>
<th>Run 3</th>
<th>Run 4</th>
<th>Run 5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. 
   5 INPUT A,B
   10 LET C = 0
   15 LET C = C + 1
   20 IF C = 40 THEN GOTO 45
   25 LET X = INT(A*RND( )) + B
   30 PRINT X,
   35 GOTO 15
   45 END

   Run this program several times. Here are some good test values to input for A and B.

<table>
<thead>
<tr>
<th>Run 1</th>
<th>Run 2</th>
<th>Run 3</th>
<th>Run 4</th>
<th>Run 5</th>
<th>Run 6</th>
<th>Run 7</th>
<th>Run 8</th>
<th>Run 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>A = 1</td>
<td>A = 10</td>
<td>A = 10</td>
<td>A = 100</td>
<td>A = 1</td>
<td>A = 10</td>
<td>A = 10</td>
<td>A = 0</td>
<td>A = 5</td>
</tr>
<tr>
<td>B = 0</td>
<td>B = 0</td>
<td>B = 1</td>
<td>B = 0</td>
<td>B = 5</td>
<td>B = 5</td>
<td>B = 5</td>
<td>B = 5</td>
<td>B = 100</td>
</tr>
</tbody>
</table>
Write a program to solve each of the following problems. Save a listing of your final solutions because you will be using these programs again.

1. Have the computer print your name at random locations along each line of output.

2. Create a number guessing game. The computer randomly selects a number between 1 and 100. The player tries to guess that secret number. The computer responds with the clues "lower" or "higher" until the player guesses correctly.

3. Create a multiplication drill game. The computer randomly selects two numbers between 1 and 10 and tells the player to multiply the two numbers. If the player types in the correct product, the computer prints "Good job... You got it!". If the player types in an incorrect product, the computer prints "Boooo!" and tells the player the correct product.

EXTRA FOR EXPERTS

Adjust your programs for problems 2 and 3 to select random numbers in any desired range.
Activity 28

EXPLORE

1. Type in each program. Record the computer's output. Explore!

A. 10 FOR X = 1 TO 5 STEP 1
   20 PRINT "GO"
   30 NEXT X
   40 END

Output:

Compare the output of this program with the output of program 1 on Activity 26.

B. 20 FOR M = 63 TO 115 STEP 1
   30 PRINT "M IS ";M,
   40 NEXT M
   50 END

Output:

2. Rewrite the program below using the FOR/NEXT concept. Run both programs and compare the output of the two programs.

10 LET K = 1
20 IF K > 15 THEN GOTO 60
30 PRINT K, K * 2
40 LET K = K + 1
50 GOTO 20
60 END

Program: Output:
Type in each program. Record the computer's output. Explore!

1. 20 FOR N = 63 TO 115 STEP 4
   30 PRINT "N IS ";N,
   40 NEXT N
   50 END

   Output:

2. 10 INPUT S
   20 FOR N = 63 TO 115 STEP S
   30 PRINT "N IS ";N,
   40 NEXT N
   50 END

   Run this program several times. Some good test values to input for S
   are 3, 0, 100, and -1.

The program below allows you to explore a FOR/NEXT loop by selecting the three values in the FOR statement. Try it with A < B, with A > B, with S positive, with S negative, with whole number values, with decimal values. Try as many different possibilities as you can think of. Be creative in your selection of input values to explore fully how a FOR/NEXT loop works. You may wish to keep a record of the values used for A, B, and S, and of the effect those values had on the FOR/NEXT loop.

3. 10 INPUT A, B, S
   20 FOR N = A TO B STEP S
   30 PRINT "N IS ";N,
   40 NEXT N
   50 END

   Notes:
   What happens when you add:
   45 GOTO 20
   What happens when you type:
   45 GOTO 10
APPENDIX E
INTERVIEW SCRIPT AND MATERIALS
INTERVIEW QUESTIONS

I. Generalization: \((10a+5)^2 = 100a^2 + 100a + 25\)

1. "I want you to look for a short-cut method for multiplying a 2-digit numbers ending in 5 by itself."

2. "What is \(15 \times 15\)"

3. "Do you see any patterns?"

4. "Can you state a rule for a short-cut?"

5. Repeat 2-4 with \(25\times25, \ldots, 95\times95\) or until s/he states a rule and cannot refine the rule.

\[
\begin{align*}
15^2 &= 225; & 35^2 &= 1225 \\
45^2 &= 2025; & 55^2 &= 3025 \\
75^2 &= 5625; & 85^2 &= 7225
\end{align*}
\]
II. Generalization: Triangle inequality

Materials: 39 lengths (3 ea. 1-13 cm), Chart to record different trials.

1. "I can make a triangle out of these 3 lengths, 6-4-3 and these, 10-10-10"
2. "Can you make a triangle with any 3 lengths?"
3. If student says NO
   a) "Tell me 3 lengths that do not make a triangle."
   b) "Why won't these form a triangle?"
   c) "Can you find a general rule for deciding when 3 lengths can form a triangle? (ask him/her to be more specific, if necessary)
   d) If s/he is wrong or vague, go to 5. If s/he is right go to 6.
4. If student says YES - "Please make a triangle with lengths 7-2-13."
5. Pick 3 length (see below)
   a) "Can these 3 lengths form a triangle?"
      -If answer wrong, show them correct answer.
   b) "Why will (won't) these form a triangle?"
   c) "Can you find a general rule for deciding when 3 lengths can form a triangle?"
   d) Repeat 5 times or until the student no longer refines or
6. "Write these numbers down. For each of these sets of numbers, state whether or not you can make a triangle."
   
   a) 100, 101, 100
   
   b) 33, 20, 56
   
   c) 8, 7, 7
   
   d) 3, 6, 3
   
   e) 19, 7, 11

7. "Find 3 lengths that will make a triangle."
   
   - Don't let student check with lengths until s/he gives numbers.

8. "Find 3 lengths that won't make a triangle."
   
   - Don't let student check with lengths until s/he gives numbers.

   6, 6, 12
   5, 3, 13
   9, 5, 3
   7, 7, 7
   9, 8, 3, 0, 13, 2
   7, 3, 4, 3, 4, 8
   3, 4, 5, 9, 13, 5
III. Generalization: Sum of interior angles of convex $n$-gon is $180(n-2)$.

Materials: drawings of various convex polygons (each numbered), sufficient number of paper wedges to measure angles, chart to keep track of number angles, sides and sum of various polygons.

1. "I want you to try to find a general rule for finding the sum of the angles of any figure. You can use these paper wedges to find the measure of each angle." Using the paper wedges, measure the angles of figure A1 (30-60-90 triangle) and calculate the sum. Be sure to try wedges that are larger than the angle (e.g. $100^\circ$) and smaller (e.g. $80^\circ$).

2. "Is this true for any triangle?"
   - If answers NO, repeat 1 with different triangle. Repeat three times and then go on to 3.

3. "What is the sum of the angles of this 4-sided figure?"
   - If student does not know, have him/her use paper wedges.

4. "Is this true for any 4-sided polygon?"
   - If answers NO, repeat 3 with different 4-sided polygon. Repeat three times and then go on to 5.

5. "Can you state a general rule for the sum of the angles of any polygon?"
a) If says YES and states correct rule then go to 8.
b) If says YES but states vague rule then go to 6.
c) If says NO then go to 8.

6. "Do you want to check your answer?" Have student pick a polygon and measure the angles.

7. Repeat 5-6 until the student no longer modifies or refines rule.

8. "Can you find the sum of the angles of a 10-sided figure? A 100-sided figure?"

<table>
<thead>
<tr>
<th>Figure</th>
<th>Angles</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>30-60-90</td>
</tr>
<tr>
<td>A2</td>
<td>20-30-130</td>
</tr>
<tr>
<td>A3</td>
<td>40-40-100</td>
</tr>
<tr>
<td>B1</td>
<td>90-90-90-90</td>
</tr>
<tr>
<td>B2</td>
<td>50-130-50-130</td>
</tr>
<tr>
<td>B3</td>
<td>30-90-150-90</td>
</tr>
<tr>
<td>C1</td>
<td>100-110-110-110-110</td>
</tr>
<tr>
<td>C2</td>
<td>140-80-120-160-40</td>
</tr>
<tr>
<td>C3</td>
<td>130-90-100-120-100</td>
</tr>
<tr>
<td>D1</td>
<td>120-120-120-120-120</td>
</tr>
<tr>
<td>D2</td>
<td>60-150-150-150-150-60</td>
</tr>
<tr>
<td>D3</td>
<td>140-100-100-160-130-90</td>
</tr>
<tr>
<td>E1</td>
<td>120-100-100-160-130-90</td>
</tr>
<tr>
<td>E2</td>
<td>90-140-130-150-100-150-140</td>
</tr>
</tbody>
</table>
Examples of Paper wedges used to measure the angles in Task Three
<table>
<thead>
<tr>
<th>Number of Sides</th>
<th>Sum</th>
</tr>
</thead>
</table>

**Task Three**
TASK THREE

Three-sided Figures

A1

A2

A3
TASK THREE
Four-sided Figures

B1

B2

B3
TASK THREE

Five-sided Figures
TASK THREE
Six-sided Figures
TASK THREE

Seven-sided Figures