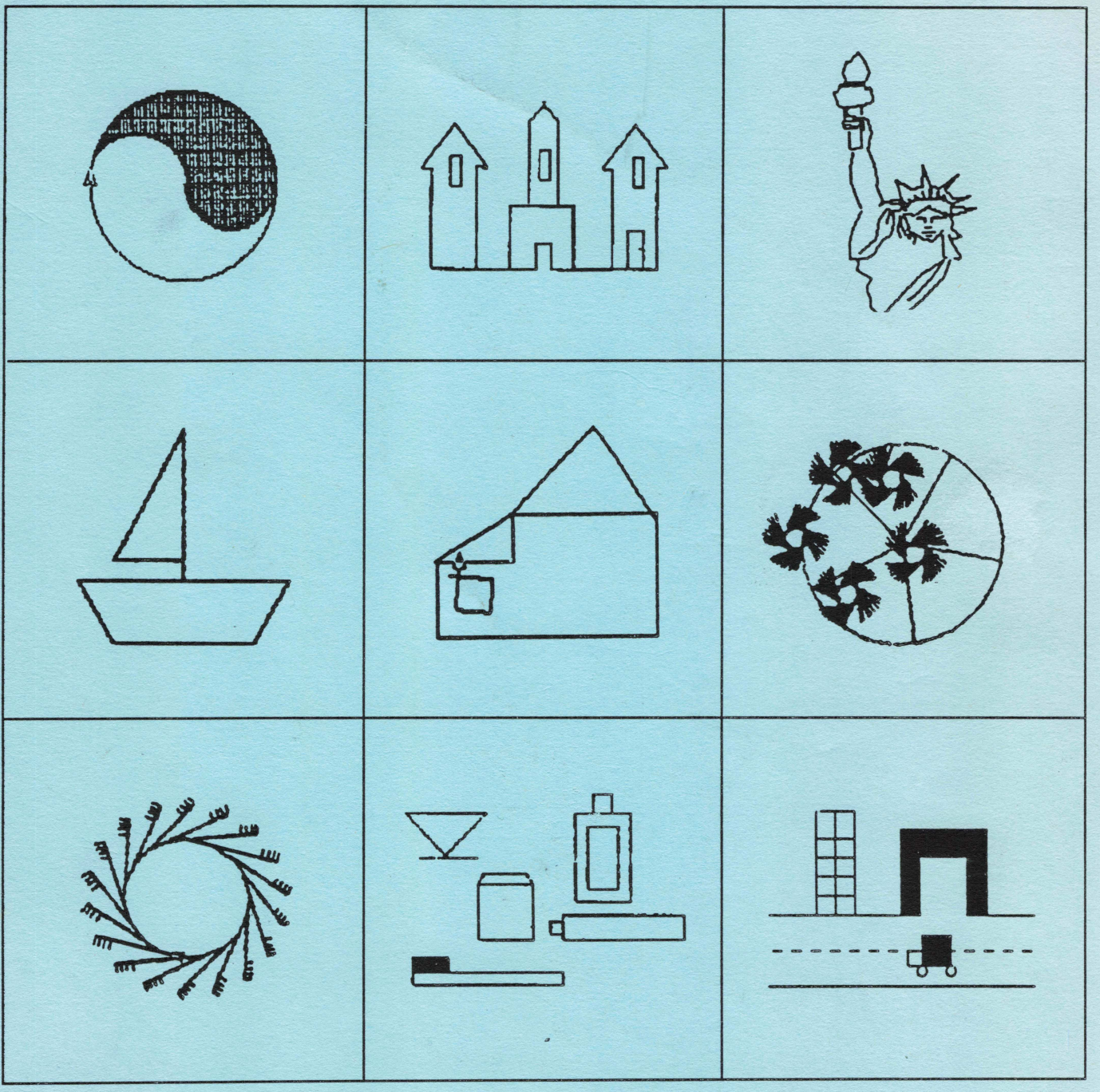
LOGO LEARNING

STRATEGIES FOR ASSESSING CONTENT AND PROCESS

DANIEL LYNN WATT AND MOLLY LYNN WATT



ISTE Publications

ABOUT THE AUTHORS

Daniel Lynn Watt and Molly Lynn Watt are deeply committed to improving teaching and learning in schools. They see teachers as a key to educational reform, and have worked to develop professional institutes, resources, and collaboratives to support teachers in deepening their understanding of the content and practice of teaching. Currently, they are senior associates at Education Development Center (EDC) in Newton, Massachusetts, where they codirect the Logo Action Research Collaborative. Together they coauthored *Teaching With Logo: Building Blocks for Learning* (Addison-Wesley, 1986) and were founders and codirectors for seven years of the Logo Institute, a summer immersion program for Logo-using educators. They are frequently invited to speak at conferences and consult with schools nationally and internationally.

Daniel Lynn Watt is a senior curriculum developer at EDC, focusing primarily on elementary mathematics. From 1976-1981 he was a research associate with the MIT Logo Group. In addition he has taught at the upper elementary, secondary, college, and graduate school levels, most recently on the education faculty at Keene State College, specializing in elementary mathematics and science eduction. He is the author of the widely used book series, *Learning With Logo* (McGraw-Hill, 1983, 1984, 1985), and has written and lectured widely on Logo and educational uses of technology.

Molly Lynn Watt directs another NSF-funded project at EDC—Action Research: Purposes, Practices, Possibilities, which is producing a monograph linking action research to national reform efforts in science mathematics and technology education. She is author of the *Welcome to Logo!* workbook series (D.C. Heath, 1985) and has written and lectured widely on educational uses of technology, action research by teachers, and authentic assessment. She has taught at all levels from preschool through graduate school.

Editor: Ron Renchler, International Society for Technology in Education
Design: Tamara Kidd, International Society for Technology in Education
Production: Lori Novak
Production Assistance: The Electronic Page

© International Society for Technology in Education, 1992
1787 Agate Street, Eugene, Oregon 97403-1923
Order Desk: 800/336-5191
Phone: 503/346-4414
Fax: 503/346-5890
CompuServe: 70014,2117
Internet; ISTE@Oregon.uoregon.edu

ISBN 1-924667-98-2

Table of Contents

Introduction 1

Chapter 1. Procedures and Subprocedures 7

Chapter 2. Procedures With Variables 13

Chapter 3. Mathematics of Turtle Geometry—Using 360 Degrees 19

Chapter 4. Mathematics of Turtle Geometry—Using Mathematical Thinking 25

Chapter 5. Strategies for Learning Through Exploration 31

Chapter 6. Debugging Strategies 37

Chapter 7. Planning, Carrying Out, and Completing a Logo Project 43

Chapter 8. Managing Workspace and Files 49

Chapter 9. A Reflective Teacher's Interventions With a Tentative Learner of Logo—A Case Study 53

Chapter 10. When Teacher *and* Student Are Stumped 59

References 63

Appendix. Critical Aspects Checklist and Notepage 65

Introduction

What do students learn when they learn Logo? And how do we, as their teachers, know what they are learning? These are the starting points for an investigation that we will be sharing together in this book. Logo was designed as an integrated learning environment in which students could learn important ideas in mathematics, computer science, and problem solving while inventing and carrying out their own computer-based programming projects. Think about your own students for a moment. What do they learn when they engage in Logo activities? Which mathematical ideas are they learning? Which ideas from computer science are they using to carry out their programming projects more effectively? Are they learning and using problem-solving strategies that can be applied to problems in other areas? This book is an attempt to help us all develop a common language and some strategies for answering such questions. It has grown out of work we have been carrying out with National Science Foundation funding to support classroom teachers in documenting Logo learning in their own classrooms.

Is Something Missing?

In response to a teacher's challenge shown in Figure I.1, Jimmy, a sixth-grade student, produced the design shown in Figure I.2. His teacher gave students the set of geometric figures shown in Figure 1.1 and challenged them to write a Logo procedure to draw one of the circular shapes. Jimmy's procedure, TO A, went beyond the teacher's challenge by filling in half of the design. His version of Logo lacked a built-in Fill command, so he had to invent his own way of filling in circles.

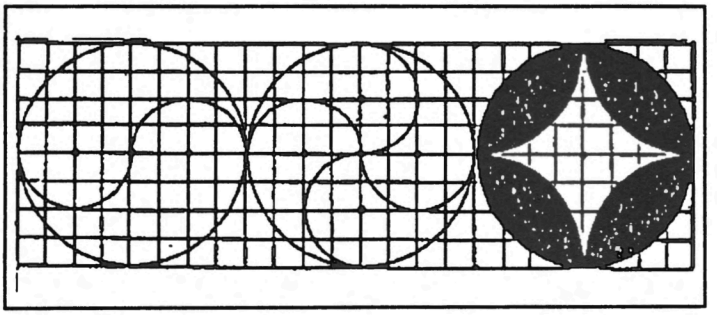


Figure I.1. Teacher's Challenge

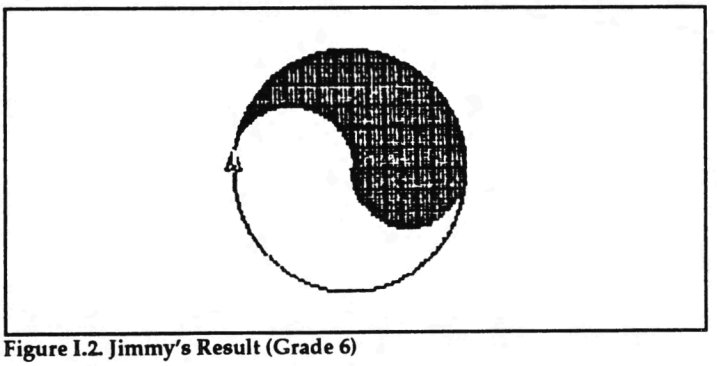


Figure I.2. Jimmy's Result (Grade 6)

```
TO A
  FULLSCREEN
  HT
  LT 90
  REPEAT 180 [FD 57.5 BK 57.5 RT 1]
  PC 0
  REPEAT 180 [FD 57.5 BK 57.5 RT 1]
  PU
  HOME
  LT 90
  FD 28.75
  PD
  REPEAT 180 [FD 28.75 BK 28.75 RT 1]
  PU
  FD 57.5
  PD
  PC 1
  REPEAT 180 [FD 28.75 BK 28.75 RT 1]
  ST
  PU
  FD 57.5
  FD 28.75
  RT 90
  PD
  REPEAT 360 [FD 1 RT 1]
END
```

It's clear from Jimmy's design and his procedure that he had learned a great deal about Logo graphics. He used the idea that a "total turtle trip" is 360 degrees and that a turtle rotation of 180 degrees can be used to fill a semicircle. And he figured out how to create "negative fills" by drawing over a previously filled region in the background color. Furthermore, Jimmy must have done some careful experimenting to determine that the radius of a circle made by REPEAT 360 [FD 1 RT 1] is 57.5 turtle steps, and applied mathematical reasoning to determine that the radii of the smaller semicircles are half that value.

But is something missing? If we look more closely at Jimmy's work, we can identify important Logo ideas that Jimmy might have used, but didn't. For one thing, he did not give his procedure a meaningful name that would help him to remember, or allow another reader to understand, what the procedure was supposed to accomplish. For another, he used the statement REPEAT 180 [FD *some distance* BK *the same distance* RT 1] four times, without using variables or creating a subprocedure. And then there is his use of the Home command. This made it easy for Jimmy to move the turtle back to the place it had started; but at the same time, it made it impossible to place the design anywhere else on the screen or to draw it in a different position.

If we were Jimmy's teachers, this little assessment would help us think about how Jimmy's Logo work could be even more powerful than it is. The next steps a teacher might take with Jimmy include helping him learn to use subprocedures with variables and to name his procedures effectively. Perhaps the most important thing for Jimmy to understand is how to write procedures that other people can read easily, so they can enjoy the elegant designs he creates and use his ideas as models for their own projects. And if Jimmy were to revise his procedure to make it more readable, he might also find ways to eliminate unnecessary steps and speed up the drawing.

Why Assess Logo Learning?

There are three major rationales for our work with classroom teachers in the assessment of Logo learning. Jimmy's example illustrates the first rationale: the need for classroom teachers to understand what their students are learning, what Logo knowledge they are actually using (and not using) in their work on a day-to-day, week-to-week basis. By knowing this, teachers can support students in using Logo as effectively as possible. The second rationale is to find ways that school districts that have been using Logo for some time can obtain specific information about what their students are learning. The third rationale is to document some of the ways in which classroom-based research can support the ongoing professional development of Logo teachers (see Watt, M., 1990; Watt &Watt, 1988).

The first rationale is extremely compelling to us. Most Logo teachers want to use Logo in the way it was intended—as an environment for open-ended, mathematical problem solving and for student-initiated projects. But in order to teach effectively when students are working on their own projects, a teacher needs to know what important skills and ideas a student might learn in working through a particular activity. The teacher needs to know which of those ideas a student is actually using at any particular time. It is also useful to know which of the skills *not* being used are especially appropriate for the student to learn next, and to know some (individualized) teaching strategies for helping the student learn those ideas.

Recall the excitement of a Logo classroom where students have discovered how to create fascinating geometric patterns that wrap around the screen. However, students often get stuck in a pattern of retyping variations of REPEAT 9999 [FD 1234 RT 5678] without particularly noticing the effects of changing the input numbers. Or think of a classroom in which students are working on complex projects. How often have you seen, for example, a potentially lovely turtle drawing of a jet plane with one of its wings upside down, and with a procedure that consists of one long sequence of FORWARDs, BACKs, RIGHTs, LEFTs, PENUPs, and PENDOWNs with no subprocedures? Creating geometric patterns and doing complex projects are appropriate as one phase of a student's Logo learning experiences, especially if students use these experiences as starting points for further learning.

However, when we see classroom work in which all the student programming is in long lines of undifferentiated code, or in which all the students are creating elaborate wrapping designs without noticing the effects of the particular numbers they are typing, we are apt to suspect that the teacher has missed too many "teachable moments." This might have occurred because the teacher did not know what important ideas the students could be learning, or because the teacher did not have a repertoire of appropriate challenges and interventions, or—as happens far too often—because the teacher focused only on the pictures that appear on the computer monitor and did not pay enough attention to what the students had been doing to make the turtle draw those pictures.

Justifying the Use of Logo

The second rationale for assessing Logo learning is equally compelling. As Logo advocates and enthusiasts, we are frequently asked by teachers and computer coordinators to help them justify their use of Logo to others in their communities. We are frequently asked if there is any research that proves that Logo students are learning important mathematical ideas, transferable problem-solving strategies, critical thinking skills, and so forth. "I need this information to convince my principal, department head, superintendent, or school board that our Logo program should be continued," the teachers tell us. Our response is to ask teachers, "What are *your* students learning? What benefits does Logo have for *them*, and how do you know?"

Unfortunately, most of the answers we received before beginning this work on Logo assessment were broad generalizations, such as "My students are learning to love mathematics," "They are solving all kinds of problems that I can't solve myself," "One of the best things for my students is that they are learning to work together and respect each other," and "Some of my 'low-ability' students get more involved with Logo than with anything else. It's so good for their self-confidence." Or sometimes teachers say, "Well, to tell you the truth, I haven't looked at their Logo work very much lately. I've got so many other subjects to teach. But I know it's valuable for them."

We believe that the positive responses many teachers give when asked about Logo learning are based on very real achievements of their students. And we think that hard observational data can probably be collected to support their generalizations. Nevertheless, we don't expect principals, school boards, or superintendents to be convinced without seeing data showing some specific learning benefits for students in real classrooms in their school districts.

On the other hand, we also believe that outside researchers have not answered these questions satisfactorily. And we believe that the best source of this data is classroom teachers who are looking at and analyzing the work of their own students. Whether the results are positive or negative, research conducted in other circumstances tells us little about what students are learning in our own classrooms.

Critical Aspects of Logo Learning

These were some of the considerations that led to the an 18-month exploratory, collaborative classroom research

project devoted to assessing Logo learning, and to the establishment of the Logo Action Research Collaborative, a follow-up project designed to support teachers in developing their own classroom research projects focused on Logo teaching and learning. Both projects were funded by the National Science Foundation, and carried out in collaboration with classroom teachers and school districts at the Education Development Center in Newton, Massachusetts.

Which aspects of Logo learning can and should be assessed by classroom teachers? Our work with teachers is focused on the upper elementary and middle school levels (Grades 4-8), and on ideas and skills involving mathematical thinking, introductory computer science, and problem solving. To focus our investigations, we invented a phrase: "Critical Aspects of Logo Learning." According to our criteria, critical aspects include behaviors, skills, and understandings that

- have been identified by Logo practitioners and researchers as essential to students' effective use of Logo
- are not usually learned by students without some form of teacher intervention
- are part of the normal learning opportunities for most Logo students
- are readily accessible to observation by classroom teachers.

As an example of what we mean by all this, let's look at the use of clearly named subprocedures, a feature that was missing from Jimmy's work shown previously in Figure I.2. Most of us would agree that if students don't eventually learn to use subprocedures well, they are missing out on one of the most powerful ideas embedded into Logo by its developers. However, many studies have shown (and our own classroom observations often confirm) that students don't necessarily learn to use subprocedures without some type of teacher intervention. Yet our experience confirms that the use of subprocedures can be learned by all but the youngest Logo learners, and can be observed by any teacher who periodically examines student procedures on screen or on paper.

Some Suggestions for Using This Book

Logo Learning—Strategies for Assessing Content and Process is designed to help Logo teachers maximize their students' learning. It is intended primarily for individuals teaching at the upper elementary and middle school levels—the grade levels at which Logo is most commonly used. Depending on your own level of Logo knowledge, you may actually learn a great deal about Logo as you read through this book. Later in this introduction, we suggest a number of other resources that can help you learn a great deal more.

To organize our observations, we have devoted each of the next eight chapters to a critical aspect of Logo learning. Each chapter has four to eight subheadings. The subheadings identify specific behaviors, skills, or concepts related to Logo learning, and are they arranged roughly in the order in which students typically develop the skills and behaviors.

Our list of critical aspects is far from a complete survey of everything students can learn by using Logo. What we attempt to do is cover the fundamentals—aspects of Logo that most teachers want and expect their students to learn in Grades 4-8—and develop a common language and common descriptions for these fundamental areas. You will also notice that we are not trying to document transfer of Logo learning to non-Logo domains. We believe it is premature to attempt to observe transfer of Logo learning until we can document what students are actually learning in Logo itself.

The critical aspect discussed in each chapter is illustrated by many examples of Logo work by students in Grades 4-8. The approximate age ranges of these students are: Grade 4, 9-10 years; Grade 5, 10-11 years; Grade 6, 11-12 years; Grade 7, 12-13; and Grade 8, 14-15 years. Their work is included to illustrate the breadth and complexity of what Logo students representing a wide range of ability and experience can actually accomplish in Logo classrooms.

Assessment is just one tool for improving the teaching and learning of Logo We believe that Logo teachers will teach more effectively if they regularly examine their students' work. To assess Logo learning properly, we believe that teachers should:

- have a good knowledge about some of the important ideas their students might be learning from Logo
- have ways to determine which of these ideas students are and are not using in their work
- have a sense of how these ideas typically develop over time
- have strategies for supporting students in making more effective use of particular Logo ideas as their work progresses from one project to another.

Teachers who use assessment as part of teaching in this way will need to set their own priorities and create their own lists of objectives. Our intention in writing this book is to share some of the ideas we identified as important in our research project and to offer some examples of student work that demonstrate how those ideas are being used.

Appendix 1 provides a Critical Aspects Checklist and Notepage. You can use the line at the left of each item to indicate the student's level of mastery, from "needs help" to "could teach this to someone else." Because we think of this as an approximate, inexact instrument, levels of mastery are indicated as positions along a continuous line rather than as fixed points. You can use the space to the right of the page for anecdotal notes about whatever seems most important.

It is your responsibility to adapt and modify the list for your own purposes. The critical aspects we have identified are not intended to be a comprehensive list of all Logo learning objectives. We encourage you to make your own list of learning objectives and to add to the items on our list. Although our list is far from complete, it is probably too long to use in assessing any particular student or group of stu-

dents. You will probably want to consider using only a subset of the list at any given time. Ideally, you should adapt the checklist for use on a word processor, which you can do quite easily on your own. (For information on acquiring a ready-made, word-processed version of the list, see Appendix 1.)

Learning to Use and Teach Logo More Powerfully

One of the great pleasures for Logophiles (those who love Logo) is that Logo presents so many opportunities to continue our own learning. The domains of mathematics, computer programming, and problem solving—and the ways Logo links them together—provide us with myriad ways to discover new content and deepen our understandings of what we already know.

Fortunately, thousands of Logo users over the past two decades have begun to create a rich and expanding supply of resource material, including books, articles, Logo-based microworlds, and teaching materials. Many of these items can support you and your students as you learn more about the aspects of Logo described in this book. Others will take you far beyond the topics we cover, to explore advanced uses of Logo or Logo applications in animation, music, language, robotics, artificial intelligence, and so forth.

The best place to start may be professional journals. Much of the material in this book was originally published in *Logo Exchange* and *The Computing Teacher*, both published by the International Society for Technology in Education (ISTE). Current and back issues of these publications will supply you with practical ideas for using Logo and teaching with it. In addition, these publications include up-to-date information about different versions of Logo, reviews of Logo books, and summaries of Logo research. Many of the articles from the early years of *Logo Exchange*, along with numerous other Logo articles and relevant background information, are collected in *Logo Theory and Practice*, edited by Dennis Harper (1989). Another publication with excellent resources for Logo teachers is the *Journal of Computers in Mathematics and Science Teaching*, published by the Association for the Advancement of Computing in Education.

When it comes to books, Seymour Papert's classic, *Mindstorms* (1980), should be in the library of every Logo-using educator. We would be remiss not mention our own books, the *Learning With Logo* series (Watt, D., 1983, 1984, 1985) and *Teaching With Logo* (Watt & Watt, 1986b). The *Welcome to Logo!* series of student and teacher books, by Molly Watt (1986), is a useful source of student activities and advice to teachers.

Logo was originally developed during the 1970s and early 1980s by the MIT Logo Group under Seymour Papert's leadership. Several excellent Logo resources have been produced by members of that group. Among them are *Turtle Geometry* by Harold Abelson and Andy diSessa's (1981); two introductory Logo reference books, *Logo for the Apple II* and *Apple Logo*, by Harold Abelson (1982a, 1982b); and *Cultivating Minds: A Logo Casebook*, by Sylvia Weir (1987).

There are so many other useful Logo books that it's hard to know where to begin. Among our personal favorites are two books by David Thornburg (1984, 1986), *Exploring Logo Without a Computer* and *Beyond Turtle Graphics*. Those who want to expand their Logo knowledge well beyond what's needed for elementary and middle school students would do well to consult *Advanced Logo*, by Michael Friendly (1988), and the three-volume series *Computer Science Logo Style*, by Brian Harvey (1985, 1986, 1987).

The Logo Action Research Collaborative

The work described in this book is a product of the Logo Action Research Collaborative (LARC), a network of Logo teacher-researchers now spanning nine sites nationwide. The project is funded by the National Science Foundation as a teacher enhancement activity. Its processes and methods, however, are largely derived from a previous NSF-supported classroom research project, "Exploratory Research on Critical Aspects of Logo Learning: A Collaborative Project With Teachers as Observers."

National studies of teacher professionalism, such as the Carnegie Report (Carnegie Forum on Education and the Economy, 1986) and the Holmes Report (Holmes Group, 1986), have strongly advocated expanded professional roles for teachers as a key aspect of school improvement. We are convinced that research by teachers is an effective avenue for professional growth. This project is an effort to provide a supportive setting for such growth.

The Logo Action Research Collaborative is creating a professional development model based on "action research" and supported by a series of collaborative research meetings conducted throughout a school year. Action research is research carried out by classroom teachers for the purpose of assessing and improving teaching and learning in their own classrooms and sharing what they have learned with other educators. We are developing a set of materials and activities to support experienced Logo teachers in carrying out action research activities with a focus on Logo learning, and we are supporting school district personnel in leading research meetings with teachers in their own districts.

Our model for supporting action research involves a one-year cycle of 14 after-school meetings with three overlapping phases. The first phase—roughly four sessions—is devoted to developing a common way of seeing and speaking. We do this by working collaboratively on Logo programming projects as a group of adults and reflecting on the knowledge we have used. We also study examples of students' work to identify the knowledge they have and have not used. During this phase we draw on our shared experiences as Logo teachers to identify what each of us means by the phrase "critical aspects of Logo." We also begin to assess whether students are using those aspects of Logo, and we begin to identify appropriate next steps for particular students.

During the second phase, teachers define and refine their own research questions and identify ways to gather data that fit in with their teaching approaches. Then they begin to gather data and attempt to make sense of it through preliminary data analyses. Although the overall guiding question of the Logo Action Research Collaborative might be stated as "What are my Logo students learning, and how can I support

them to learn as much as they can from their Logo experiences?," participating teachers have investigated a range of other questions that arise out of their own specific teaching objectives and circumstances. For example, they have investigated some of the following questions:

- How can I support my students to use procedures and subprocedures more effectively?
- How can I teach my fourth-grade students to use variables and recursion?
- How can I help my students collaborate more effectively when they work together on Logo projects?
- Since I have to assign grades for Logo projects, can I set grading criteria that evolve throughout the year and involve my students in assessing their own ongoing work?
- What do students who use Logo on the Macintosh accomplish that is different from what students who use Logo on Apple II computers accomplish?

During this second phase of the research process, participants examine each other's data, describe what they see, look for patterns in the data, help to clarify each other's research focus, and deal with issues generated by the process of the research itself.

The third phase of the cycle is devoted to a more thorough analysis of the data each teacher has gathered over the course of the year. Part of our meeting time is spent supporting participants in identifying findings in their data and in writing reports about what they have learned.

During the 1989-90 school year, we began working with a group of teacher-researchers who teach in Grades 4-8 in Boston, Brookline, and Concord, Massachusetts. During 1990-91, the project expanded to involve nine different school districts and a broader range of grade levels. The group of teacher-researchers from Boston, Brookline, and Concord are led by past project participants. The six additional groups are led by experienced Logo teachers and teacher educators in Bellevue, Washington; Brattleboro, Vermont; Chapel Hill, North Carolina; Chicago, Illinois; Ladue, Missouri; and Madison, Wisconsin. During the summer of 1990, group leaders from all nine districts attended a one-week leadership institute at the Education Development Center in Newton, Massachusetts. Teachers at the nine sites are now involved in testing our model for supporting teacher research. Our personal role is now that of administrators, facilitators of group leaders, and documenters of the project.

Our ultimate goal is to develop and disseminate a process that can be carried out by groups of teachers with local leadership. Our work will result in a one-week orientation and planning institute for leaders, a handbook and guidelines for leaders, a Logo assessment guidebook for teacher-researchers, a monograph of teachers' research reports on Logo learning, and an assessment of the project that will support its replication in other settings. We are particularly interested in collaborating with other educators to apply the methods we have developed to content areas other than Logo.

We have written and will continue to write extensively about the processes and outcomes of the Logo Action Research Collaborative (Watt, D., 1989; Watt, M., 1989a, 1989b; Watt & Watt, 1988, 1989, 1990). Also, we have devoted Chapters 9 and 10 of this book to case studies of the work of two teachers in our research groups. These two case studies illustrate some of the collaborative processes used in our research meetings and some of the research strategies used by teachers. More particularly, they show how teachers in our research groups have used ideas like the ones discussed in this book to assess their students' work, and how this has led to improved teaching and learning in their classrooms.

Acknowledgments

The work that this book is based on was very much a group effort. Most of the examples come from our first cycle of classroom research carried out in the spring of 1987. Our research assistant for that work was Joan Funk, and our collaborators were nine computer coordinators and classroom teachers from four Greater Boston communities. These teachers included Elizabeth Berrner, David Crump, Diana Freedman, Doreen Kelly, Robert Kondel, Jane Manzelli, Mary Miller-Teehan, Pasquale Puleo, and Kitsy Rothermel. All of these individuals contributed to the development of the ideas and material we will be sharing here, and they collected most of the illustrative examples of student work. Additional examples came from a follow-up project carried out by Logo teachers in Boston, Brookline, and Concord during the 1989-90 school year. These teachers included Gerri Abrams, Lorita Brockington, David Crump, Peter Lavelle, Jim MacNeil, Judith Randolph, Donna Rosenberg, and Patricia Rowe.

We also want to thank our colleagues and support staff at the Education Development Center. Glenn Kleiman and Myles Gordon, as directors of EDC's Center for Learning Technology, gave us critical support in writing and carrying out the proposals that supported this work. Angela Daskalos, Cynthia Grzelcyk, and Gerri Sills provided essential behind-the-scenes administrative support. Project advisors Betty Bjork, Douglas Clements, Claryce Evans, Jan Hawkins, Judith Kull, Roy Pea, and Sylvia Weir helped shape our thinking at early stages. Our editors, Sharon Yoder of *Logo Exchange* and Anita Best of *The Computing Teacher*, gave us encouragement by editing and publishing earlier versions of this material as separate articles in their respective journals.

Our ideas for supporting classroom action research grew out of our own experiences as teacher-researchers and from a growing literature about such research (Watt & Watt, 1982). Our extensive work with the development of teacher centers encouraged us to value teachers' professionalism. We modeled some of our seminar structures on the Children's Thinking Seminars developed and led by Bill and Sara Hull (Hull, 1979; Watt, M., 1979). Some of our ideas about supporting teacher investigations were learned from the Teacher Initiated Research Project led by Claryce Evans (Evans, Stubbs, Duckworth, & Davis, 1981), and from An Experiment in Teacher Development, led by Jeanne Bamberger, Eleanor Duckworth, and Maggie Lampert (Bamberger, Duckworth, & Lampert, 1981; Duckworth, 1987). In our research and

development work with Patricia Carini at the Prospect Center in North Bennington, Vermont, we participated in developing procedures for teachers to support each other's classroom action research projects (Carini, 1979). Over the years, our participation in the North Dakota Study Group on Evaluation has provided many models of qualitative research, observation, and documentation processes teachers can carry out in their own classrooms. Other references to the literature on classroom research are included in the bibliography.

Finally, we want to acknowledge the substantial support and encouragement we received from our program managers at the National Science Foundation. Raymond Hannapel, of NSF's Research in Teaching and Learning Program, and Christine Stevens and Joan Ferrini-Mundy, of NSF's Teacher Enhancement Program, played an essential role in shaping and supporting our work. This work was supported primarily by NSF grants MDR 8651600, Exploratory Research on Critical Aspects of Logo Learning, and TPE 8855541, Professional Development of Logo Teachers Through Classroom Research Seminars.

Although the work of our collaborators was essential to developing the content of this book, the authors take full responsibility for the particular ways these ideas are expressed and organized. They do not necessarily reflect the views of any of our collaborators, the Education Development Center, or the National Science Foundation.

Chapter 1
Procedures and Subprocedures

Levels of Critical Aspect 1
Procedures and Subprocedures

A. Preprocedures

B. Using REPEAT

C. Sequential, Undifferentiated Procedures

D. Use of Subprocedures

E. Structured Programming Styles

F. Modular Programming Styles

G. Procedures Using Recursion and Conditionals

Procedural Thinking—The Importance of "Mind-Sized Bites"

Essentially, a procedure is a recipe, a step-by-step set of instructions designed to achieve a specific result. Logo allows us to give procedures any names we want and allows any procedure to call any other procedures as subprocedures to carry out any task. This is Logo's core strategy for helping us solve problems. In its simplest form, this means we can divide a large task into smaller parts, give each part its own name, and work on each small piece of a larger task individually. Of course each subprocedure, once named, can call on still other subprocedures to solve smaller parts of the problem, and so on. Or, alternatively, if we have a procedure already defined for a particular purpose, for example, to draw a small square, we can use the same procedure again for a different purpose, for example, to draw the window of a house.

In *Mindstorms*, Seymour Papert (1980) quotes a seventh grader, Robert, who had discovered the usefulness of this approach: "See, all my procedures are mind-sized bites.... I used to get mixed up by my programs. Now I don't bite off more than I can chew." Dividing problems into mind-sized bites allows us to limit and identify the small piece of a problem we are working on at any given time. Giving each piece an understandable name allows us to come back to find any particular piece again when we want it, to reuse it when appropriate, and to debug larger problems more effectively by identifying the exact subprocedure in which a bug occurs.

All of this makes a lot of sense to us as teachers—but it doesn't always make sense to students. Many teachers and researchers report that their students don't use procedures and subprocedures effectively in solving problems (Hillel & Samurcay, 1985; Leron, 1983; MacDougall, 1986). Jimmy, whose procedure TO A was used in the Introduction (see Figure I.2), is a classic example of a student who could have used Logo much more powerfully if he had used clearly named procedures and subprocedures in his work. Part of our job as Logo teachers is to see that our students learn these ideas—not as abstract principles that they are made to memorize for a test, but as useful techniques that can make their own work easier and more understandable for themselves and others (Papert, DiSessa, Watt, & Weir, 1979; Watt D., 1979a, 1979b; Watt & Watt, 1986b).

This last point is both subtle and important. In any creative work, there are always at least two audiences: the creator, and other people who may be interested in both appreciating and using that work (or some of the ideas in it) for themselves. As teachers, one of our overriding tasks is to create a community of learners who can build on the ideas others create and who can contribute ideas for others to use (Watt & Watt, 1986b). The importance of this may be temporarily lost on a student (or on any of us) in the midst of a struggle to solve a particularly thorny problem. But part of the genius built in to Logo is that when a Logo procedure is clearly written, that is, with appropriately defined and clearly named subprocedures, anyone can read it, even someone with less Logo knowledge than its creator. As teachers we can help students keep this in mind as they revise and

complete their work. Just as writers may need many drafts to communicate their ideas as effectively as possible, most Logo programmers, even the most experienced, usually have to revise their work to make it more understandable for themselves and others (Watt & Watt, 1987a).

In our work with teachers, we have identified seven different levels of sophistication in using procedures and subprocedures. However, students don't always learn to use procedures in the particular order we have set out and may not always function consistently at one level. For example, we have noticed that many students learn to write clear, well-articulated superprocedures while still using sequential programming style for all their subprocedures. Or a student may use one or two procedures as tools without really understanding modular programming. Or students may learn to use tail recursion in its simplest form before understanding much about other aspects of procedures and subprocedures.

Here are some descriptions, illustrated by examples of student work, of the different levels we've identified for learning to use procedures and subprocedures.

A. Preprocedures

Preprocedures involve using the *idea* of a procedure without actually creating one, for example, repeating the steps needed to draw a square several times without defining a square procedure, or using a series of commands over and over in a procedure without defining a particular subprocedure. The learner is clearly identifying a sequence of commands that is used more than once for a particular purpose but does not choose to (or does not know how to) define a procedure or subprocedure for that purpose.

Even students who have learned to define procedures often include many examples of preprocedures within long sequential procedures. When we see students working in this way, we take it as a sign of their readiness for more formal procedural thinking (or for using REPEAT) and take it upon ourselves to show (or remind) them how to use a procedure or a REPEAT command to simplify their work.

B. Using REPEAT

For some students, using REPEAT may be another form of preprocedure. Although it does not usually involve naming, using REPEAT often first occurs in situations where students realize that a particular set of instructions can be used repeatedly to produce an effect more easily than by typing a long sequence of commands. For example, it is easier to draw a square by typing REPEAT 4 [FD 50 RT 90] in one line than by typing eight separate commands. Beginning Logo students often use REPEAT to produce dramatic or unexpected effects, without necessarily noticing either the number of REPEATs or the sequence of commands being repeated. For example, using commands such as REPEAT 10000 [FD 1234 RT 4567] does not necessarily lead to procedural thinking!

Brad, a fourth-grade student, wrote the procedure shown in Figure 1.1, Brad's CITY, which demonstrates how he used REPEAT.

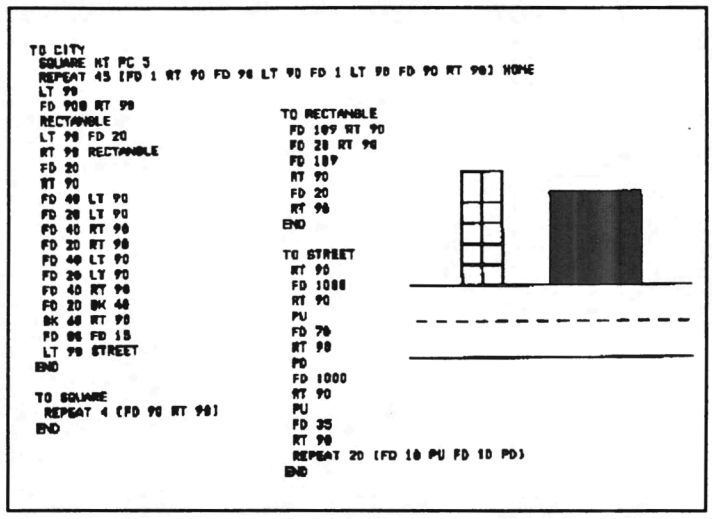


Figure 1.1. Brad's CITY (Grade 4)

Brad's CITY procedures illustrate three different uses of REPEAT. Brad used REPEAT in his SQUARE procedure; he also used it in unconventional ways to fill in a square (beginning with the line REPEAT 45 in CITY) and to draw a dotted line (beginning with REPEAT 20 in STREET). Much of Brad's other work consisted of explorations using REPEAT in many different ways. (For further discussion of this idea, see Chapter 5, "Strategies for Learning Through Exploration.")

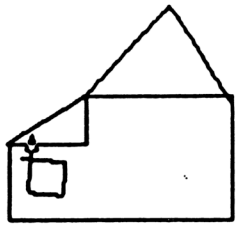
C. Sequential, Undifferentiated Procedures

This involves taking a sequence of commands that produced a desired effect in direct mode, giving them a name, and typing them into the editor. This mode is characterized by long command lines, with many commands to a line, and by frequent typing errors that occur while typing commands from notes or memory. Students may use idiosyncratic names for procedures: TO A, TO ME, TO FLUFFY.

Figure 1.2, Kathy's HOUSE.6, provides an example by a fourth-grade student. Kathy's HOUSE.6 procedure is a classic sequential, undifferentiated procedure. She seems to have constructed her house in direct mode, step by step, using a trial-and-error approach and writing down each command she used. When she was convinced that a series of commands achieved the desired effect, she would add the commands to her already lengthy procedure. Her work includes some elements of preprocedures as well: she uses PENUP [*move the turtle*] PENDOWN sequences twice to place the turtle in position without drawing. This indicates that she might be ready to write separate MOVE procedures to accomplish this, which would also have the effect of dividing her house into parts (frame, roof, window, and so forth).

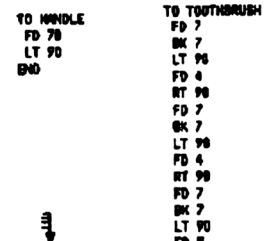
A later project by Kathy, shown in Figure 1.3, Kathy's TOOTHBRUSH, also illustrates sequential, undifferentiated programming, as well as the use of preprocedures.

Although Kathy has two separately named subprocedures, HANDLE and TOOTHBRUSH, both of them must be typed to draw the figure shown. Notice that she uses the sequence FD 7, BK 7, LT 90, FD 4, RT 90 repeatedly, without using



```
TO HOUSE.6
 PU
 BK 30
 BK 20
 RT 90
 FD 30
 FD 20
 LT 90
 PD
 FD 50
 FD 10
 FD 20
 LT 90
 FD 65
 FD 30
 LT 90
 FD 30
 RT 90
 FD 30
 FD 20
 LT 90
 FD 30
 FD 10
 FD 5
 FD 4
 LT 90
 FD 56
 FD 56
 FD 20
 FD 10
 FD 3
 PU
 LT 90
 FD 56
 FD 30
 BK 5
 BK 4
 BK 3
 LT 45
 RT 5
 RT 3
 RT 4
 PD
 FD 20
 FD 30
 FD 20
 FD 5
 LT 103
 FD 80
 RT 15
 FD 60
 PU
 FD 4
 LT 90
 LT 29
 LT 10
 FD 20
 LT 10
 LT 5
 BK 5
 PD
 FD 10
 FD 8
 FD 5
 FD 2
 FD 4
 RT 90
 FD 6
 FD 4
 FD 5
 FD 7
 RT 90
 FD 10
 FD 7
 FD 3
 FD 3
 RT 90
 FD 3
 FD 2
 FD 2
 FD 5
 FD 5
 FD 6
 FD 3
 FD 1
 FD 1
END
```

Figure 1.2. Kathy's HOUSE.6 (Grade 4)



```
TO HANDLE
 FD 70
 LT 90
END
```

```
TO TOOTHBRUSH
FD 7
BK 7
LT 90
FD 4
RT 90
FD 7
BK 7
LT 90
FD 4
RT 90
FD 7
BK 7
LT 90
FD 5
RT 90
FD 7
BK 7
LT 90
RT 90
FD 7
BK 7
RT 90
LT 180
FD 4
END
```

Figure 1.3. Kathy's TOOTHBRUSH (Grade 4)

either REPEAT or a subprocedure. This is characteristic of undifferentiated programming, and it is also a nice example of the use of preprocedures within an existing procedure. (Readers might be interested to know that by the time Kathy wrote this procedure, she had been taught and encouraged to use subprocedures and had used them effectively with her teacher's assistance in other projects. But here she shows clearly that she has not yet mastered their use.) Notice also the literal incorporation of typing errors and errors made in direct mode into the procedure: using FD 5 rather than FD 4 (we're *guessing* this was an error, given the regularity of a toothbrush); using LT 90 RT 90 without FD 4 between them; and using RT 90 LT 180 without replacing them by LT 90. Once again, these are characteristic of the sequential, undifferentiated mode of Logo programming. It is not easy for Kathy to debug her work! Chapter 9 provides a case study of how Kathy's teacher worked with her over a period of several months to support her in using Logo more powerfully.

D. Use of Subprocedures

The use of subprocedures begins by giving separate names to procedures that draw different designs or accomplish different parts of a task. Names begin to reflect purposes: TO TRIANGLE, TO HEAD, TO LEFTLEG. Some students continue to work sequentially, giving separate names to each part of a drawing. Others create shapes separately and make more complex designs by rotating and combining them. Students do not always distinguish between the parts of a procedure that draw a design and the parts that move the turtle into place. Therefore, interface bugs (parts of designs out of place) are common when subprocedures are first used.

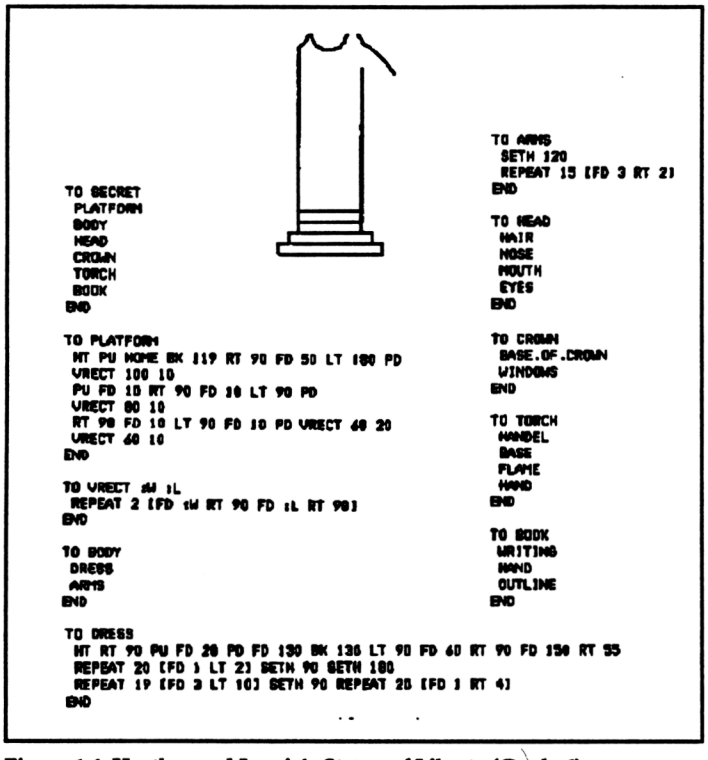
Brad's CITY procedure (see Figure 1.1) shows an early stage of development in a series of related projects that were extended and elaborated for a period of months while Brad also worked on other projects. Brad's work illustrates some use of subprocedures and modularity (see Section F in this chapter), mixed with sequential programming. For example, Brad has used the subprocedures SQUARE, RECTANGLE, and STREET to draw these elements of his city. But he has not used subprocedures to identify either of the two within his CITY procedure or to separate the parts of his pictures from the commands that place them on the screen. Nor does he use a subprocedure or REPEAT for the repeated sequence of commands that draws the criss-cross design on the taller building. Modularity is hinted at here by the reuse of the same subprocedure, RECTANGLE, in two places. Modular programming showed up a lot more in Brad's later CITY projects. (We have already commented on Brad's mastery of the REPEAT command.) A more consistent use of subprocedures is illustrated in the next section.

E. Structured Programming Styles

Structured programming usually involves planning. Students decide in advance what parts will be needed for a given project and choose names that reflect those parts. Naming becomes more thoughtful and complex. Procedures that move the turtle may be defined and named separately from procedures that draw parts of designs. A structured programmer may start with a fairly definite plan, but the plan (and naming strategies) are usually modified in the course of

work on a project. Some programmers do minimal planning in advance; instead, they create a program sequentially and then impose a structure in revising the project after it is partially or totally working.

The turtle drawing in Figure 1.4, Heather and Joanie's Statue of Liberty, illustrates the beginning of structured programming. (The example given here shows just one stage of the work in progress. We'll come back to this project in Chapter 7, "Planning, Carrying Out, and Completing a Logo Project.")



```
TO SECRET
 PLATFORM
 BODY
 HEAD
 CROWN
 TORCH
 BOOK
END

TO PLATFORM
 HT PU HOME BK 119 RT 90 FD 50 LT 180 PD
 VRECT 100 10
 PU FD 10 RT 90 FD 10 LT 90 PD
 VRECT 80 10
 RT 90 FD 10 LT 90 FD 10 PD VRECT 60 20
 VRECT 60 10
END

TO VRECT :W :L
 REPEAT 2 [FD :W RT 90 FD :L RT 90]
END

TO BODY
 DRESS
 ARMS
END

TO DRESS
 HT RT 90 PU FD 20 PD FD 130 BK 130 LT 90 FD 60 RT 90 FD 150 RT 55
 REPEAT 20 [FD 3 LT 21 SETH 90 SETH 180
 REPEAT 19 [FD 3 LT 10] SETH 90 REPEAT 28 [FD 1 RT 4]
END

TO ARMS
 SETH 120
 REPEAT 15 [FD 3 RT 2]
END

TO HEAD
 HAIR
 NOSE
 MOUTH
 EYES
END

TO CROWN
 BASE.OF.CROWN
 WINDOWS
END

TO TORCH
 HANDEL
 BASE
 FLAME
 HAND
END

TO BOOK
 WRITING
 HAND
 OUTLINE
END
```

Figure 1.4. Heather and Joanie's Statue of Liberty (Grade 6)

The girls' superprocedure and subprocedure names illustrate the use of a top-down problem-solving style and a structured programming style. Modularity (discussed in Section F of this chapter) is shown by the use of VRECT, a variable rectangle procedure borrowed from an earlier project. The separation of the command lines in PLATFORM into lines that move the turtle with its pen up and lines that draw parts of the picture are a further indication of an awareness of modular programming styles. Both the PLATFORM and DRESS subprocedures, however, indicate that sequential, undifferentiated procedures are still very much part of Heather and Joanie's approach to Logo. They do not seem to be thinking about whether another reader can readily understand how their procedures work or whether they themselves might have difficulty debugging them someday.

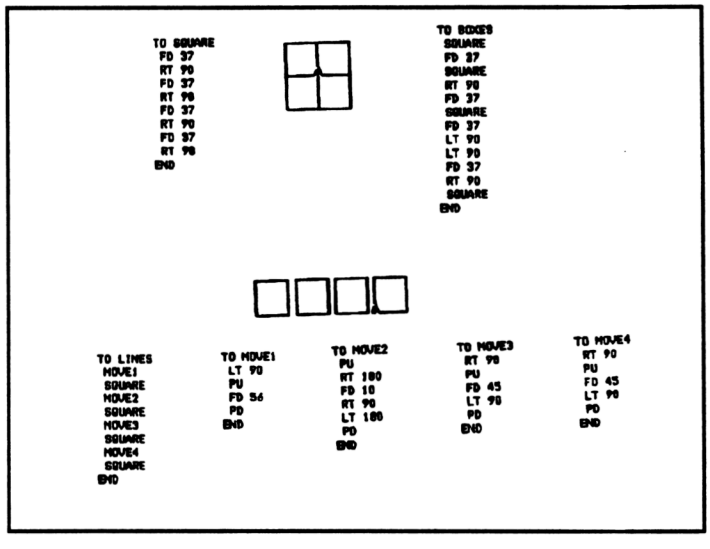
A Note About Problem-Solving Approaches

Students' use of procedures and subprocedures are closely related to their preferred problem-solving approaches. We can talk about two major problem-solving approaches, "top down" and "bottom up," although most of us use some combination of these in any actual work. Top-down styles involve starting with a specific goal, breaking a problem into parts, and solving it systematically, part by part. Bottom-up styles involve building up a complex structure bit by bit without necessarily knowing in advance what the eventual result will be. Many students have a preference about which approach to emphasize in their problem solving, and this shows up in their Logo work. In *The Second Self*, Sherry Turkle (1984) calls these preferences "hard mastery" and "soft mastery" and has some excellent illustrations of the power of both approaches used in Logo. We think that Brad's work in Figure 1.1 typifies what Turkle calls "soft mastery," while Heather and Joanie used a "hard mastery" approach in Figure 1.4. We discuss planning and completing projects at more length in Chapter 7. For now, we'd like to point out that it is more common to use combinations of these styles in solving a particular problem than it is to use either of them in their purest form.

F. Modular Programming Styles

Modular programming involves the use of the same subprocedures in more than one setting, for example, the use of a circle procedure as a tool for drawing a particular design, or the creation of a design with interchangeable parts (a face that can have different sets of features). Sometimes procedures made for one purpose can be used for another. It is wise to make modular procedures state-transparent (see Chapter 3, Section E) so that they can be used in any position or for any purpose. Modular procedures with variable inputs are more flexible and powerful than procedures with fixed inputs.

Figure 1.5, Andy's BOXES and LINES, offers examples of modular and structured programming styles.



```
TO SQUARE
 FD 37
 RT 90
 FD 37
 RT 90
 FD 37
 RT 90
 FD 37
 RT 90
END

TO BOXES
 SQUARE
 FD 37
 SQUARE
 RT 90
 FD 37
 SQUARE
 FD 37
 LT 90
 LT 90
 FD 37
 RT 90
 SQUARE
END

TO LINES
 MOVE1
 SQUARE
 MOVE2
 SQUARE
 MOVE3
 SQUARE
 MOVE4
 SQUARE
END

TO MOVE1
 LT 90
 PU
 FD 56
 PD
END

TO MOVE2
 PU
 RT 180
 FD 10
 RT 90
 LT 180
 PD
END

TO MOVE3
 RT 90
 PU
 FD 45
 LT 90
 PD
END

TO MOVE4
 RT 90
 PU
 FD 45
 LT 90
 PD
END
```

Figure 1.5. Andy's BOXES and LINES (Grade 5)

Andy's superprocedures show modularity clearly. In each project, the same procedure, SQUARE, is used to draw four different squares. Notice, however, that modularity may be subservient to structured programming in Andy's thinking. For instance, although MOVE3 and MOVE4 are identical procedures, Andy gives them different names, which show

their order in the overall plan. (And if Andy's MOVE1 procedure had left the turtle pointing straight up, all of the next three MOVE procedures could have been the same, which would have been an even stronger example of modularity.)

G. Procedures Using Recursion and Conditionals

It is not clear where these fit in the hierarchy. They have a range of sophistication of their own: simple tail recursion, recursion with conditional stop rules, recursion with incrementing variables, and embedded recursion. Few students in Grades 4-8 use embedded recursion. We say a bit more about recursion in Chapter 2, when we deal with variables.

Some Closing Thoughts

Each of these examples, as well as Jimmy's "yin-yang" design shown in the Introduction (Figure I.2), illustrates work in progress typical of many children's Logo work in upper elementary grades. Each example shows how the students used some of Logo's powerful ideas involving procedures and subprocedures, and the examples also show how they ignored or only partially utilized some potentially useful ideas.

What we have tried to show here is that there is a lot to be learned by taking a detailed look at student work in process. We leave it to you to think about what you would suggest to Kathy, Brad, Heather and Joanie, Andy, or Jimmy if you were their teacher. Specific decisions about how and when to intervene with each child depend on a teacher's best insights about the specific knowledge that would be most useful and productive in helping individual students meet their own immediate goals. Teachers will also have to consider which knowledge best meets their long-range learning goals for their Logo students and what types of teaching intervention work best for each individual student.

Chapter 2
Procedures With Variables

Levels of Critical Aspect 2
Procedures With Variables

A. Prevariables

B. Procedures With Simple Inputs

C. Procedures With Multiple Inputs

D. Using Inputs With Subprocedures

E. Procedures With Proportional Variables

F. Procedures Involving Recursion, Incrementing Variables, and Conditionals

G. Using Global Variables (MAKE)

Variables—Using Names to Manipulate Information

Understanding the use of Logo variables is a major step in harnessing the computer's power. Logo variables provide experiences with some of the most important ideas in mathematics. For example, using Logo variables to draw shapes of different sizes provides an intuitive introduction to the idea of a mathematical *function*, a set of mathematical rules or operations whose result (output) depends on one or more inputs. The use of Logo variables with mathematical operations also paves the way for understanding the algebraic concept of an *unknown*, the notion that we can give a name or label to a piece of information, manipulate that name or label symbolically without knowing exactly what it is, and substitute a specific value later. This concept is one of the prerequisites for understanding almost any branch of mathematics.

At the same time, using variables unlocks the power of the computer to carry out *algorithms*, or repetitive tasks in which the same operations are applied to information that changes as the task repeats until some condition is satisfied. This is one of the core ideas underlying all computer science. Another important computer science and problem-solving concept supported by using Logo variables is the idea of *modularity*—the use of one procedure or tool for more than one purpose.

A Logo variable has one word of any length as a name and can have a number, word, or list as its value. Logo variables come in two flavors, *local variables*, which are defined only within a particular procedure and all its subprocedures, and *global variables*, which can be used within any procedure or at the direct command level.

Beginning Logo programmers usually learn to define procedures to make the turtle draw simple shapes on the screen. Local variables can be used to generalize such a procedure, that is, to give one procedure the capability of drawing many different shapes. Most of the students whose teachers collaborated in our research used variables primarily in this way: local variables, defined within particular procedures to represent numbers. Hillel and Samurcay (1985) have studied the processes by which students learn to generalize their procedures using variables.

Logo variables can be given arbitrary names, just like Logo procedures. As with procedures, students may need encouragement from a teacher to move from using idiosyncratic or ritualistic naming strategies to using meaningful names that can communicate the purpose of each variable to a reader of their procedures.

Some students in elementary grades work with Logo without ever using variables. Some teachers may be reluctant or feel unprepared to teach their students to use variables. However, we found a number of students in Grades 4 and higher using variables on their own without direct instruction. They may have learned to use variables by working with other Logo users outside of class, by reading books, or by analyzing tool procedures. We have identified several

different levels of sophistication involved in using ideas about variables, but we want to emphasize that students do not necessarily learn to use variables in this order. Clements (1988-1989) has written an excellent series of articles that review and summarize Logo research about how students learn, and don't learn, to use Logo variables.

A. Prevariables: Using the Idea of Variables Without Using Inputs or Variable Names

Students show that they are ready to use Logo variables when they create projects in which one number or parameter is varied consistently. For example, a student may define several procedures to draw squares of different sizes and use procedures that are identical in every respect except the input to FORWARD, or a student may use REPEAT with different angle inputs to create different shapes. For example, the TRIANGLES and SPIRALS procedures in Figure 2.1—Andy's Exploratory Work—show that he is using the idea of variables without actually knowing the Logo syntax for writing procedures with inputs. This is a good example of what we mean by prevariables.

```
TO TRIANGLES
 REPEAT 3 [FD 50 RT 120]
 REPEAT 3 [FD 75 RT 120]
 REPEAT 3 [FD 100 RT 120]
END

TO SPIRALS
 REPEAT 36 [FD 1 RT 10]
 REPEAT 36 [FD 2 RT 10]
 REPEAT 36 [FD 3 RT 10]
 REPEAT 36 [FD 4 RT 10]
 REPEAT 36 [FD 5 RT 10]
 REPEAT 36 [FD 6 RT 10]
END
```

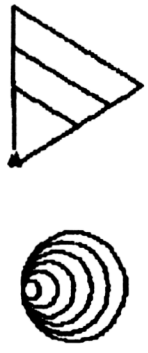


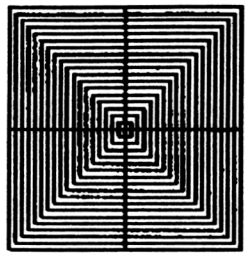
Figure 2.1. Andy's Exploratory Work (Grade 5)

For his SPIRALS procedure, Andy could have been shown how to use a subprocedure such as

```
TO CIRCLE :STEP
  REPEAT 36 [FD :STEP RT 10]
END
```

rather than a sequence of REPEATs. This would help him begin to use variables directly.

Sam's DESIGN, shown in Figure 2.2, also illustrates prevariable thinking. Sam uses a variable SQUARE procedure as a subprocedure of DESIGN, but he uses it in very much the same way Andy used FORWARD in Figure 2.1.



```
TO DESIGN
 SQUARE 90
 SQUARE 85
 SQUARE 80
 SQUARE 75
 SQUARE 70
 SQUARE 65
 SQUARE 60
 SQUARE 55
 SQUARE 50
 SQUARE 45
 SQUARE 40
 SQUARE 35
 SQUARE 30
 SQUARE 25
 SQUARE 20
 SQUARE 15
 SQUARE 10
 SQUARE 5
END
```

```
TO SUPERDESIGN
 DESIGN
 LT 90
 DESIGN
 LT 90
 DESIGN
 LT 90
 DESIGN
END
```

```
TO SQUARE :LENGTH
 REPEAT 4 [FD :LENGTH RT 90]
END
```

Figure 2.2. Sam's DESIGN (Grade 6)

B. Procedures With Simple Inputs

The simplest procedures with variables are ones that produce fixed shapes of different sizes. These procedures are usually constructed by starting with fixed-size procedures and then substituting a variable name for the size in place of fixed sizes. Sam's SQUARE :LENGTH procedure in Figure 2.2 is a typical example of this. Such procedures are often given to students as models or as tools to use in creating more elaborate designs than they could make with procedures that draw only shapes of one size.

C. Procedures With Multiple Inputs

This type of procedure may include shapes for which more than one size is varied. An example occurs in the VRECT procedure shown in Figure 2.3, Heather and Joanie's VRECT :W :L procedure, in which :W and :L are meant to stand for width and length, respectively. VRECT was used as a tool to produce part of the platform on which the statue was to stand in the girls' Statue of Liberty project (see Figure 1.4).

```
TO PLATFORM
 HT PU HOME BK 119 RT 90 FD 50 LT 180 PD
 VRECT 100 10
 PU FD 10 RT 90 FD 10 LT 90 PD
 VRECT 80 10
 RT 90 FD 10 LT 90 FD 10 PD VRECT 60 20
 VRECT 60 10
END

TO VRECT :W :L
 REPEAT 2 [FD :W RT 90 FD :L RT 90]
END
```



Figure 2.3. Heather and Joanie's VRECT :W :L Procedure (Grade 6)

Heather and Joanie used VRECT with fixed inputs to create the drawing they wanted. The variable names they used indicate a clear understanding of what the rectangle procedure and its variables are supposed to do. (In Chapter 7, we discuss Heather and Joanie's Statue of Liberty project more fully.)

The following procedures written by Jerry, a fourth grader, illustrate a more sophisticated use of procedures with more than one variable. He started out by writing a procedure to make a Logo circle:

```
REPEAT 360 [FD 1 RT 1]
```

He generalized this to a procedure with two variables to control the size of the FORWARD step and the amount of turn:

```
TO CIRCLE :SIZE :TURN
  REPEAT 360 [FD :SIZE RT :TURN]
END
```

Then Jerry realized that with turns larger than RT 1, he did not need to repeat his instruction 360 times, so he created another procedure with a variable to control the number of REPEATs:

```
TO S.M. :TIMES :SIZE :ANGLE
  REPEAT :TIMES [FD :SIZE RT :ANGLE]
END
```

Jerry's work shows how some students can easily extend their use of variables from one to several, once they understand the basic Logo syntax. However, a teacher's intervention could have helped Jerry make another leap of understanding by helping him see that the computer could do the calculations he was doing in his head.

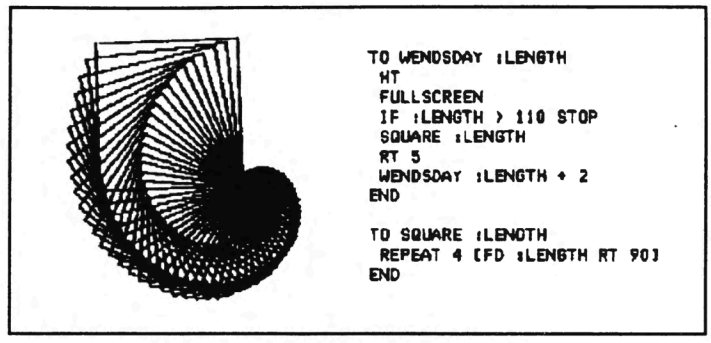
In addition, this would have allowed him to build an important mathematical relationship directly into his procedure. Jerry explained that he knew how to find the angle for a particular shape by dividing 360 by the number of sides. But he had not yet figured out that he could incorporate this arithmetic directly in his procedure, for example, by writing

```
TO POLYGON :TIMES :SIZE
  REPEAT :TIMES [FD :SIZE RT 360/:TIMES]
END
```

D. Using Inputs Within Subprocedures

These procedures can take on a variety of forms: complex drawings in which all elements vary together (using TRI :SIZE and SQ :SIZE to build HOUSE :SIZE), or procedures that use different sizes of the same shape (varying squares to produce a tower or a tunnel). Students typically proceed from using variable procedures with fixed inputs, as in Sam's work (Figure 2.2) and Heather and Joanie's projects (Figure 2.3), to using variable names within subprocedures. As these procedures become more complex, it becomes important to understand how local variables work in Logo.

Tony's WENDSDAY Design, shown in Figure 2.4, follows from Sam's idea of using one procedure to make squares of different sizes and gradually increasing the sizes.



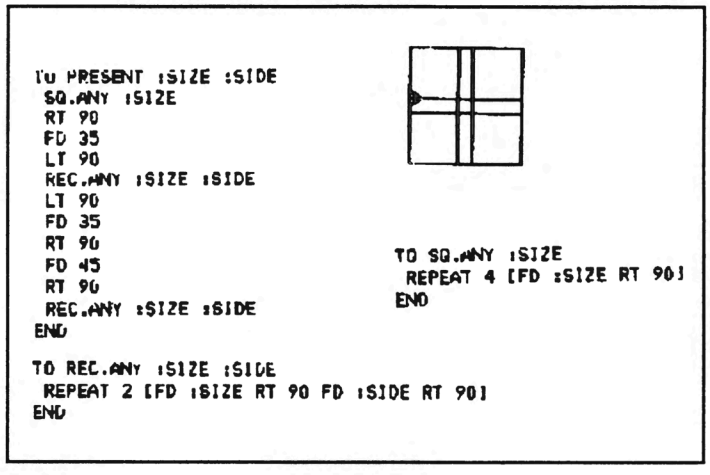
```
TO WENDSDAY :LENGTH
 HT
 FULLSCREEN
 IF :LENGTH > 110 STOP
 SQUARE :LENGTH
 RT 5
 WENDSDAY :LENGTH + 2
END

TO SQUARE :LENGTH
 REPEAT 4 [FD :LENGTH RT 90]
END
```

Figure 2.4. Tony's WENDSDAY Design (Grade 6)

Tony uses SQUARE :LENGTH as a variable subprocedure within a recursive procedure that is programmed to stop when the size variable grows larger than a given amount.

Aaron's Project in Figure 2.5 shows a less complete understanding of the use of subprocedures with variables and the use of procedures with more than one variable.



```
TO PRESENT :SIZE :SIDE
 SQ.ANY :SIZE
 RT 90
 FD 35
 LT 90
 REC.ANY :SIZE :SIDE
 LT 90
 FD 35
 RT 90
 FD 45
 RT 90
 REC.ANY :SIZE :SIDE
END

TO SQ.ANY :SIZE
 REPEAT 4 [FD :SIZE RT 90]
END

TO REC.ANY :SIZE :SIDE
 REPEAT 2 [FD :SIZE RT 90 FD :SIDE RT 90]
END
```

Figure 2.5. Aaron's Project (Grade 5)

His PRESENT procedure uses two subprocedures: SQ.ANY to draw a square, and REC.ANY to draw rectangular stripes across the square. (Aaron's teacher taught him to add ".ANY" to the name of the procedure to show that it can make a shape of any size. This is a useful convention for students beginning to use variables.) However, Aaron's PRESENT superprocedure combines fixed FORWARD steps and subprocedures with variable inputs in the same procedure. It only draws the design Aaron wanted when given the inputs 80 10. Aaron has not taken proportionality into account. His variable names, :SIZE and :SIDE, also show that although he has learned the rule for naming Logo variables, he has not clearly distinguished the different purposes of the two different variables he is using.

E. Procedures With Proportional Variables

These are procedures with variables that are proportional to the original values used in a fixed-value procedure. For a simple example, suppose we start with a procedure for a fixed rectangle, with a width three times its length:

```
TO RECT
  REPEAT 2 [FD 20 RT 90 FD 60 RT 90]
END
```

We might generalize this proportionally using only one input so that RECT 20 produces the original rectangle:

```
TO RECT :SIZE
  REPEAT 2 [FD :SIZE RT 90 FD 3 * :SIZE
    RT 90]
END
```

Or, alternatively, we might add a scale factor, so that RECT 1 produces the original rectangle, RECT 2 is twice as big, and so forth:

```
TO RECT :SCALE
  REPEAT 2 [FD 20 * :SCALE RT 90
            FD 60 * :SCALE RT 90]
END
```

This type of procedure may help students learn to understand ratios, proportion, and similarity, as well as demonstrate to them how those mathematical ideas are used for practical purposes.

Diane's SAILBOAT project illustrates the approach that uses variable lengths that are proportional to the original fixed lengths. Her goal was to make a design with several sailboats of different sizes. Her teacher suggested this project to give students experience with variables and proportionality, topics that were part of the sixth-grade math curriculum. First Diane made a sailboat using fixed lengths: 50, 150, 100, and so forth. Then she divided each length by the first length to establish the proportions needed. Then she rewrote her original procedures as shown in Figure 2.6.

```
TO SAILBOAT :L
 HT
 HULL :L
 MOVE :L
 SAIL :L
END

TO HULL :L
 LT 30
 FD :L
 RT 120
 FD :L * 3
 RT 120
 FD :L
 RT 60
 FD :L * 2
END

TO MOVE :L
 PU
 RT 90
 FD :L * 0.87
 RT 90
 FD :L
 PD
 LT 90
END

TO SAIL :L
 FD :L * 2
 LT 150
 FD :L * 2
 LT 120
 FD :L
END
```

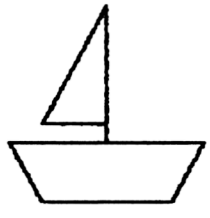


Figure 2.6. Diane's SAILBOAT Project (Grade 6)

To calculate the length factors used in her HULL, MOVE, and SAIL subprocedures, Diane had to first choose a reference length and calculate the ratio of each length in the design to that reference length. Adapting a fixed-length SAILBOAT procedure to make a variable one gave Diane an opportunity to use ratio and proportion as well as to use Logo variables in a powerful way.

F. Procedures Involving Recursion, Incrementing Variables, and Conditionals

These procedures are used when a variable changes in a repeated process. Many students get into this type of situation with recursive designs that grow indefinitely. For example, the following procedure produces a design similar to Sam's DESIGN (Figure 2.2), but the design grows indefinitely without stopping:

```
TO GROWSQ :SIZE
  SQUARE :SIZE
  GROWSQ :SIZE + 5
END
```

A conditional statement or stop rule is needed to terminate the growth:

```
TO GROWSQ :SIZE
  IF :SIZE > 100 [STOP]
  SQUARE :SIZE
  GROWSQ :SIZE + 5
END
```

Tony's WENDSDAY Design (Figure 2.4) illustrates a common variation of this, in which a rotation is added along with each increase in size.

In Section C of this chapter, "Procedures With Multiple Inputs," we have already seen that Jerry was very adept at using variables. He also demonstrated an unusually creative use of recursive programming technique. His teacher gave him a tool procedure to make the computer "waste time":

```
TO WAIT :TIME
  IF :TIME = 0 STOP
  WAIT :TIME - 1
END
```

Jerry modified this procedure on his own in the following way:

```
TO DECIDE :NUMBER
  IF :NUMBER = 1 PR [GOOD!]
  IF :NUMBER = 2 PR [AWESOME!]
  DECIDE :NUMBER - 1
END
```

He noticed that with an input greater than 1, this procedure would print both GOOD! and AWESOME! but would never stop. He then modified it as follows:

```
TO DECIDE :NUMBER
  IF :NUMBER = 1 PR [GOOD!] STOP
  IF :NUMBER = 2 PR [AWESOME!] STOP
  DECIDE :NUMBER - 2
END
```

Jerry then explained that DECIDE would print GOOD! if it started with an odd number and AWESOME! if it started with an even number:

```
DECIDE 19
GOOD!
DECIDE 24
AWESOME!
```

Jerry's work shows his flexibility in modifying a procedure that was given to him for one purpose—causing a delay or wait in one of his programs—and using the same idea for a very different purpose—to distinguish between odd and even numbers. It also shows that when teachers provide well-written tool procedures for their students to *use*, the same tools can help students learn new programming techniques.

G. Using Global Variables (MAKE)

Only a few students used global variables in our research project. Most of these students were working independently on advanced projects, and their examples are too complex to include here. Readers are referred to Abelson (1982) for a clear and complete description of how the two types of variables work and the distinctions between them. Many other Logo texts discuss this issue as well.

Some Closing Thoughts

Most students in Grades 4-8 who use variables at all have a very incomplete understanding of the topic. In some cases they pick up ideas about variables outside of class, from other students, from books, or from examining tool procedures. Some fourth-grade students (for example, Jerry) are capable of using variables quite flexibly, although their teachers do not teach variables in any formal way and may have incomplete knowledge about Logo variables themselves. Other students, such as Tony or Aaron, may learn to use variables for a particular purpose but have difficulty generalizing to more flexible uses of variables.

All of this makes variables a tricky topic to handle at these grade levels. There are many ways to approach it. An ideal Logo teacher would have a working mastery of many different ways of using variables in Logo and should be able to help each student learn to use variables in a way that best supports what that student is trying to accomplish. A more practical approach for many teachers may be to have some sample variable procedures handy and then give them to students who are ready to experiment with them and use them as models for their own work. This is what Jerry's teacher did with the WAIT procedure. A third approach is to create specific lessons or activities designed to teach the use of variables in a particular way. This is what Aaron's, Heather and Joanie's, and Diane's teachers did. Whichever approach works best for you is the one to use. However, be sure to look carefully at what your students are actually doing with variables so that you can support them in consolidating their incomplete knowledge and using it effectively to carry out specific projects.

Chapter 3
Mathematics of Turtle Geometry—Using 360 Degrees

Levels of Critical Aspect 3
Mathematics of Turtle Geometry—Using 360 Degrees

A. Understanding the Relation: Rotation x Number of Sides = 360 for a Regular Polygon

B. Distinguishing Between Complete and Incomplete Rotations

C. Recognizing and Using Special Angles (Factors of 360 and Their Multiples)

D. Using the Total Turtle Trip in Constructing Figures

E. Using State-Transparent Shape Procedures as Modules for Larger Projects

Getting to Mathland

Remember Mathland, Seymour Papert's vision of "a context which is to learning mathematics what living in France is to learning French"? By "speaking mathematics" to the turtle and solving geometric problems in an environment with math-speaking peers and teachers, Logo users have the opportunity to learn mathematics in the process of drawing pictures and designs with the turtle. Logo's turtle geometry provides a natural entry point into the exploration and use of important mathematical ideas.

Many schools use Logo to foster computer literacy, problem solving, creativity, and personal expression—all highly appropriate educational objectives. But how often is Logo used as a math activity? All too often we observe that the mathematical understandings needed to use Logo as powerfully as possible are missing from Logo students' work. In this chapter we'll look at some mathematical ideas involving 360 degrees; in the next chapter we'll consider some of the other mathematical ideas that arise in turtle geometry.

Here is an example of what we mean. Have you ever seen a "square" like the one shown in Figure 3.1?

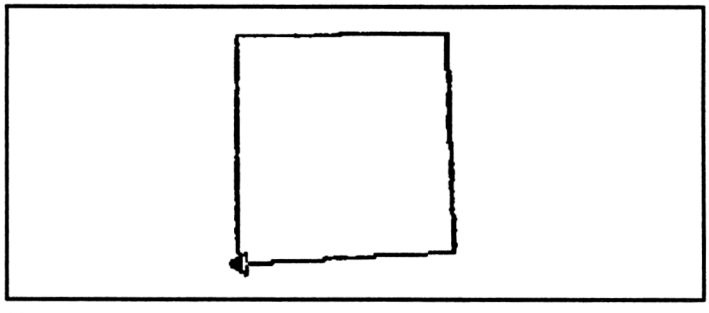


Figure 3.1. Square Example

We remember wincing when a third-grade teacher we'll call Henry proudly showed us that his students had discovered how to draw a square using a rotation of 89 degrees and then used that idea all year in their other drawings. Henry was an outstanding teacher who had embraced the "Logo philosophy" enthusiastically by allowing his students the freedom to discover their own ideas. He supported their efforts by accepting and praising whatever they created. "I never correct my students in their Logo work," he explained. "They'll come up with a 90-degree square when the 89-degree square no longer serves their purposes." But they went through an entire year without ever finding a need for another type of square or being asked by their teacher to see if they could find one!

We winced because Henry had missed an important teaching opportunity. The process of drawing a simple Logo square could have been a terrific starting point for his students to explore and eventually understand two of the most useful ideas in turtle geometry: the importance of the total turtle trip—a rotation of 360 degrees—and the usefulness of rotations that involve factors of 360, such as 90 degrees. Henry's students may never use an 89-degree rotation again, but a 90-degree rotation could have been an extremely powerful tool for them, leading to many other successful Logo

projects. Understanding the rotational symmetries involved in the total turtle trip and the interesting modular arithmetic of the 360-degree cycle opens the door to much more powerful use of Logo and to some of the core mathematical ideas of geometry, trigonometry, physics, and other disciplines using mathematics.

Why is 360 such an important number? Why didn't the mathematicians who invented degree measurements select a nice round number, such as 100 or 1000, to be the number of units in a complete rotation? We've read that 360 degrees originated with the ancient Sumerian astronomers, who estimated the length of a year—the number of days for the Earth to complete one full orbit or circle around the sun—to be 360 days. But we think this value has been a standard for thousands of years because of another property: 360 has more integer factors than almost any other number less than 1000.

In our work, we have identified five different levels in the use of 360 degrees. They are arranged roughly in the order in which many students learn to use them.

A. Understanding the Relation: Rotation x Number of Sides = 360 for a Regular Polygon

Again, the square comes first. Students easily see that a square is made up of four 90-degree turns. Some students extend this immediately to predict that a triangle would have three 120-degree turns; a hexagon, six 60-degree turns; and so on. A classic Logo circle, REPEAT 360 [FD 1 RT 1] is another well-known example of this principle. Students who don't discover this "Rule of 360" on their own can readily learn it through directed exploration. The same reasoning can be applied to draw complete spinning designs, that is, designs that repeatedly rotate a given shape through a fixed angle until the turtle is back where it started. After students have learned the Rule of 360, a nice collection of special angles—all factors of 360—becomes available for them to use in other activities.

Don't assume that a student who has learned a few cases has really mastered this idea. Andy, a fifth grader, did a lot of exploring with geometric designs and used triangles and squares for many of them. When he worked on other polygons, he had difficulty generalizing a rule, although he was attempting to discover one:

```
TO PENTAGON
  REPEAT 5 [BK 50 RT 72]
END

TO HEXAGON
  REPEAT 6 [FD 60 RT 82]
END
```

The "rule" Andy seemed to be trying in this case was: "Add 1 to the number of REPEATs and add 10 to the distance and angle."

By contrast, Jerry, whose work with variables we described in Chapter 2, had a good understanding of the Rule of 360. He wrote this variable procedure to draw polygons:

```
TO S.M. :TIMES :SIZE :ANGLE
  REPEAT :TIMES [FD :SIZE RT :ANGLE]
END
```

He explained that he could always find the angle input by dividing 360 by the number of sides!

B. Distinguishing Between Complete and Incomplete Rotations

It is important to distinguish between a complete shape, that is, a shape in which the turtle draws a figure and returns to its original state (heading as well as position), and an incomplete shape, in which the turtle draws a figure but does not return to its initial state. A technical term for a complete shape is a *state-transparent shape*—a shape that ends with the turtle in the same heading and position in which it started. A good test for a complete shape is to invoke the same command twice. If the turtle draws the second shape exactly on top of the first, the procedure is state transparent. Otherwise it is not.

Many students begin by drawing incomplete figures—the turtle returns to its starting position but not to its original heading. For example, the procedure in Figure 3.2 draws a square but leaves the turtle rotated 90 degrees from where it started, leaving an incomplete square.

```
TO SQUARE1
  FD 100
  RT 90
  FD 100
  RT 90
  FD 100
  RT 90
  FD 100
END
```

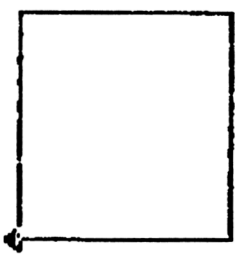


Figure 3.2. Incomplete Square

The procedure in Figure 3.3 draws a complete square.

```
TO SQUARE2
  FD 100
  RT 90
  FD 100
  RT 90
  FD 100
  RT 90
  FD 100
  RT 90
END
```

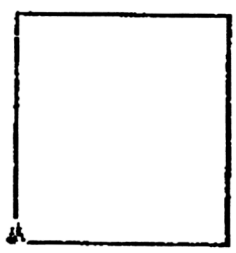


Figure 3.3. Complete Square

Another procedure that draws a complete square is

```
TO SQUARE3
  REPEAT 4 [FD 50 RT 90]
END
```

It is important to help students recognize the difference between complete and incomplete shape procedures and to see the usefulness of procedures such as SQUARE2 or SQUARE3, which can easily be used as modular pieces in other projects.

Complete rotations are also important in producing spinning designs in which a shape is rotated over and over until it produces a closed shape. Once again, knowledge of the significance of 360 degrees can be extremely helpful. For instance, Andy repeatedly rotated his TRIANGLE procedure LT 5 until he got the closed shape shown in Figure 3.4, Andy's LOTS Design. He asked his teacher to print out a dribble file of this session for him. (The class was using IBM Logo, and the teacher was collecting dribble files of student work as one form of record keeping.)

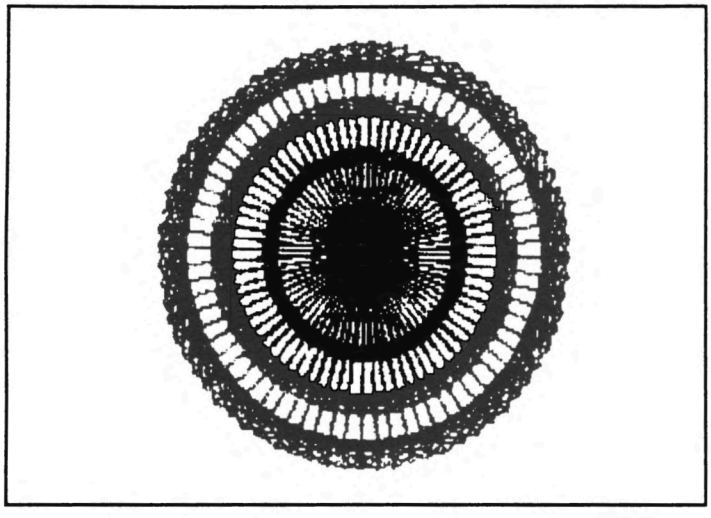


Figure 3.4. Andy's LOTS Design (Grade 5)

Andy then counted the number of rotations and came up with the following procedure, which is not complete but looks complete with the turtle hidden:

```
TO LOTS
  REPEAT 70 [TRIANGLE LT 5]
END
```

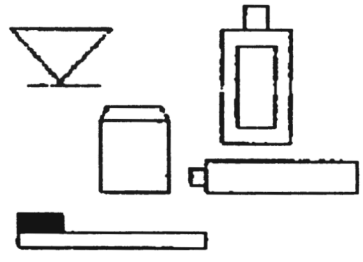
Andy's teacher might have been able to help him learn the Rule of 360 by asking him to make the turtle return to its exact starting point. One way to do this is to use the command PRINT HEADING before and after drawing the figure. If Andy had done this, he might have realized that his LOTS procedure rotated the turtle a total of 350 degrees; he might have then changed the number of REPEATs to 72 to make a total rotation of 360 degrees.

Alternatively, the teacher might have asked Andy to multiply the number of REPEATs by the rotation and compare the result to 360. Multiplying 70 by 5 also gives 350. Both strategies—printing the turtle's heading, and multiplying the number of rotations by the angle of rotation—are excellent strategies for helping students learn the Rule of 360.

C. Recognizing and Using Special Angles (Factors of 360 Degrees and Their Multiples)

Students commonly begin by using 90 degrees to make a square. Many of them generalize from this concept and learn to use 90-degree turns to move the turtle easily around the screen. Students begin to use other special angles in a variety of ways: using 180 to reverse the turtle's direction; using 45 as half of 90; combining turns of 30 and 60 degrees to make a right angle; finding that 30, 60, and 90 go together well in making drawings; adding 60 and 60 to get 120, or 90 and 45 to get 135; using 120, 60, and 45 to draw regular polygons; and so on. Eventually some students may realize that most of these angles are factors of 360. Even if they don't discover this for themselves, students who use combinations of special angles can create turtle designs more easily than students who have to work out all their angles by trial and error. There's no reason why a teacher shouldn't seize the opportunity to show students that the special angles they are using have this interesting relationship to 360 degrees.

Figure 3.5, Sheila's CUP and FLOSSY Drawings, illustrates one use of special angles.



```
TO CUP
 LT 45 FD 30 RT 45 RT 45 RT 45 FD 67 RT
45 RT 45 RT 45 FD 50 RT 45 FD 20 BK 50
END

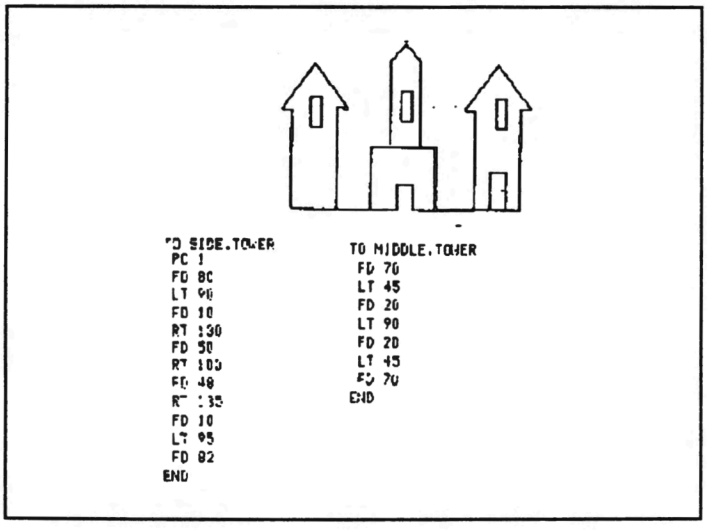
TO FLOSSY
 SEW FD 45 RT 30 FD 10 RT 60 FD 20 FD 15
 RT 60 FD 10
END

TO SEW
 REPEAT 4 [FD 45 RT 90]
END
```

Figure 3.5. Sheila's CUP and FLOSSY Drawings (Grade 4)

Sheila used 45-, 90-, 30-, and 60-degree angles in her project. Although she did not seem to know about 135 degrees, she knew how to get the turtle heading in the direction she wanted by repeating RT 45 three times. She also used rotations appropriately of 30 and 60 degrees to make a 90-degree turn in making the top of her dental floss pack.

In his Castle Project, shown in Figure 3.6, Jerry seemed to know something about using special angles but had not completely sorted out how to use them effectively.



```
TO SIDE.TOWER
 PC 1
 FD 80
 LT 90
 FD 10
 RT 130
 FD 50
 RT 105
 FD 40
 RT 135
 FD 10
 LT 95
 FD 82
END
```

```
TO MIDDLE.TOWER
 FD 70
 LT 45
 FD 20
 LT 90
 FD 20
 LT 45
 FD 70
END
```

Figure 3.6. Jerry's Castle Project (Grade 4)

Two of Jerry's subprocedures, SIDE.TOWER and MIDDLE.TOWER, show how he used angles to draw roofs on his towers. In SIDE.TOWER, the turtle started by turning LT 90, and then made a total rotation of 365 degrees: RT 130, RT 100, RT 135. (A simpler set of rotations might have been 135, 90, and 135, if Jerry had known how to use them.) Jerry compensated for the extra 5 degrees by turning the turtle LT 95 to get it heading straight down. (We can't possibly determine from reading the procedure whether he figured this out by trial and error or by mathematical reasoning.) In MIDDLE.TOWER, however, Jerry omitted the eaves and limited the turtle to left turns of 45, 90, and 45 degrees, leaving the turtle positioned correctly for a downward trip to complete the tower.

Many students use special angels without necessarily realizing their connection to 360. A teacher can help build the connection by reminding students that angles such as 30, 45, 60, and 120, which can be used to build regular polygons, are also useful for making specific drawings of nonregular shapes.

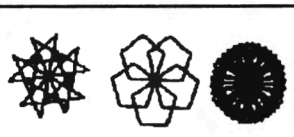
D. Using the Total Turtle Trip to Solve Problems Involved in Constructing Figures

In its fullest sense, the concept of the total turtle trip has two parts:

1. Every time the turtle makes a journey of any kind and returns exactly to its starting position and heading, it will have turned through a total rotation of exactly 360 degrees or a multiple of 360 degrees (possibly including 0 degrees).
2. If you repeat a fixed set of turtle commands indefinitely, the turtle will eventually return to its starting point, having made a total turtle trip.

Students who understand these two ideas can use them in their work in a variety of ways. Some designs become easier to complete by first calculating the total angle the turtle has already turned and then subtracting that amount from 360 degrees to see how much it must still turn. In other cases, students can use these concepts to figure out how many REPEATs they need to complete a figure. In combination, these two ideas provide a powerful reasoning tool that can be used to solve complex problems that otherwise have to be solved through trial and error.

Two projects by Martha and Vinh show how these two students developed the idea of the total turtle trip through a project their teacher assigned. They then used the idea on their own in an independent project. First, their teacher taught them how to make the supershapes shown in Figure 3.7, Martha and Vinh's Supershapes, by using regular polygons and stars.



```
TO SSTAR
 RT 45
 REPEAT 5 [STAR RT 144]
END

TO STAR
 REPEAT 5 [FD 30 RT 144]
END
```

```
TO SPENT
 LT 135
 REPEAT 5 [PENT RT 72]
END

TO PENT
 REPEAT 5 [FD 20 RT 72]
END

TO SQ1
 REPEAT 4 [FD 20 RT 90]
END

TO SSQ1
 REPEAT 36 [SQ1 RT 10]
END
```

Figure 3.7. Martha and Vinh's Supershapes (Grade 5)

In a later project, they applied the same idea (on their own!) to make a flag with 13 stars, shown in Figure 3.8, Martha and Vinh's Flag Project.



```
TO FLA
 REPEAT 13 [M2 STAR]
END

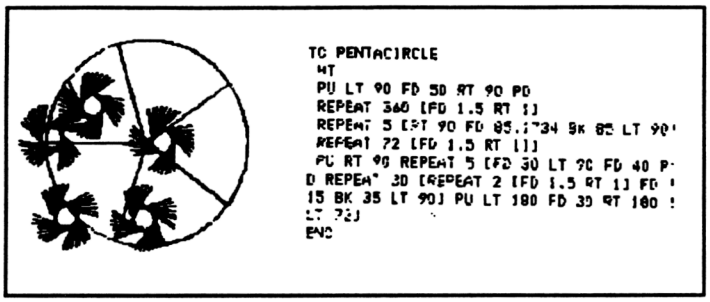
TO M2
 PU LT 27.6 BK 7 PD
END

TO STAR
 REPEAT 5 [FD 6 RT 144]
END
```

Figrue 3.8. Martha and Vinhs' Flag Project

When their teacher asked how they decided to rotate the turtle 27.6 degrees in the M2 procedure that is used in FLA, they told her that they had divided 360 by 13! Here we have an example of students applying what they had learned in an assigned project, Supershapes, to a student-generated project by using the Rule of 360!

Douglas' PENTACIRCLE, shown in Figure 3.9, was one of the classic examples in our research project. Douglas gave this project to his teacher with the following note: "(I wish that the design would go in each part of the circle.)"



```
TO PENTACIRCLE
HT
PU LT 90 FD 50 RT 90 PD
REPEAT 360 [FD 1.5 RT 1]
REPEAT 5 [RT 90 FD 85.[734 BK 85 LT 90
REPEAT 72 [FD 1.5 RT 1]]
PU RT 90 REPEAT 5 [FD 30 LT 90 FD 40 P
D REPEAT 30 [REPEAT 2 [FD 1.5 RT 1] FD
15 BK 35 LT 90] PU LT 180 FD 30 RT 180
LT 72]
END
```

Figure 3. 9. Douglas' PENTACIRCLE (Grade 4)

Douglas clearly knew a lot about using 360 degrees and the total turtle trip, because he used it in several contexts in this project. He used REPEAT 360 [... RT 1] to draw a circle, and REPEAT 5 [... REPEAT 72 [... RT 1]] to divide the circle into five equal parts. He even used REPEAT 5 [. . . *lots of commands, including two nested REPEATS* . . . LT 72] to draw his design and repeat it five times. So what was Douglas's problem? And how could his teacher help him? For the moment we leave these questions for you to explore for yourself. Douglas' work is discussed at more length in Chapter 10.

E. Using State-Transparent Shape Procedures as Modules for Larger Projects

A state-transparent shape procedure draws a shape and leaves the turtle with its position and heading unchanged. SQUARE2 , shown in Figure 3.3, is an example of such a shape. Although it sometimes takes more forethought to construct a state-transparent shape, it is much easier to use such shapes as part of a larger project than to use shapes like SQUARE1.

For example, if you want to place a lot of squares in different positions around the screen, a state-transparent procedure lets you clearly separate the procedures that move the turtle from place to place from the procedure that draws each part of the drawing. With this approach, you never have to think about where the turtle is going to end up in order to plan the next move.

Most students do not discover this technique for themselves. You'll have to teach it to them and show them why it's a good idea for their own work, or they'll never use it. Regular polygons and spinning designs such as the ones discussed earlier in this chapter provide good introductory examples of state-transparent procedures.

A Note About Modular Arithmetic and the Logo Turtle

The Logo turtle always has a heading between 0 and 360 degrees. If it rotates more than 360 degrees, Logo subtracts multiples of 360 from the total rotation to place the turtle in its new position. For example, RT 60 and RT 420 produce exactly the same effect, a rotation of 60 degrees to the right. However, it is difficult for students to learn this because the turtle does not actually rotate on the screen. It simply redraws the turtle at a new heading calculated by the computer. Students who repeatedly issue commands with large numbers, such as REPEAT 1000 [FD 1000 RT 1000], have no way of observing that a rotation of RT 1000 is the same as RT 280 (1000 - 360 - 360).

Similarly, students do not automatically discover that RT 280 has exactly the same effect as LT 80 (360 - 280) unless it is pointed out to them directly or indirectly (through a structured exploration). One approach we have used (Watt & Watt, 1986b) is to substitute teacher-made procedures that slow down the turtle for the conventional RIGHT and LEFT commands. This allows students to see the turtle actually rotate rather than just snap from one position to another.

For More Information

Many Logo books discuss the role of 360 degrees in constructing regular polygons. The following references go significantly beyond what we have covered on the topics discussed in this chapter. Chapter 1 of *Turtle Geometry*, by Abelson and diSessa (1981), includes formal proofs of theorems related to the total turtle trip and 360 degrees. "Turtle Geometry, a Mathematics Made for Learning," Chapter 3 of *Mindstorms*, by Papert (1980), describes the importance of this knowledge in solving turtle geometry problems. A number of other sources contain useful examples of ways to support students in learning about the significance and usefulness of 360 degrees in the turtle's world (Martin & Bearden, 1983; Papert, diSessa, Watt, & Weir, 1979; Watt, D., 1983a, 1983b; Watt & Watt, 1986a, 1986b).

Chapter 4
Mathematics of Turtle Geometry—Using Mathematical Thinking

Levels of Critical Aspect 4
Mathematics of Turtle Geometry—Using Mathematical Thinking

A. Estimating Angles and Distances

B. Combining Steps

C. Using Inverses: RIGHT/LEFT, FORWARD/BACK

D. Using Specific Knowledge About Particular Shapes

E. Using RIGHT/LEFT Symmetry

F. Using Similarity

G. Using Transformational Geometry

H. Using and Understanding Wrapping

Using Mathematical Thinking

What do we mean by "using mathematical thinking"? As Logo-using educators, we have always found this question to be one of the most difficult to answer. Much of what is done with Logo is inherently mathematical. Drawing even the simplest shape with the turtle involves using precise language, numbers, measures of angles and distances, geometric knowledge and relationships, and so on. But in order to teach Logo effectively, it's important to help students recognize mathematical thinking when they are using it and to show them appropriate mathematical thinking strategies when they are *not* using them on their own.

Webster's New Collegiate Dictionary defines mathematics as "the science of numbers and their operations, interrelations, combinations, generalizations, and abstractions, and of space configurations and their structure, measurement, transformations, and generalizations." Thus, when we talk about using mathematical thinking in Logo, we are talking about making use of conceptual tools involving operations, interrelations, measurements, generalizations, transformations, and so on. In a very general way, these are also the conceptual tools of ordinary school mathematics—arithmetic, geometry, algebra, and the like. Logo provides a domain in which students can begin to use these tools naturally and comfortably to solve problems of interest to them rather than to mechanically carry out calculations in a workbook.

However, to say that students can learn to use certain intellectual tools while using Logo does not mean that all students *will* learn to use them or that they will apply ideas used in one setting to other settings in which they might be useful. There are many mathematical ideas embedded in Logo. A first step for teachers helping students to acquire useful mathematical thinking strategies is to be aware of what those strategies might be and when they might be useful for particular projects. The second step is to observe whether students are using them. Finally, a teacher can utilize specific teaching and intervention strategies to help students learn and use particular ideas. Chapter 3 of *Mindstorms* (Papert, 1980) illustrates a number of the most useful and important mathematical ideas built into Logo by its developers. Other mathematical ideas are identified in the *Final Report of the Brookline Logo Project* (Papert, DiSessa, Watt, & Weir, 1979).

Chapter 3 of this book focused on how students use some of the mathematical ideas related to the 360-degree modularity of turtle geometry. In this chapter we describe a few more specific ways that the students involved in our collaborative research project used mathematical thinking in their Logo work.

As an example, Figure 4.1, Kathy's HOUSE.6, shows how Kathy drew the frame of a house.

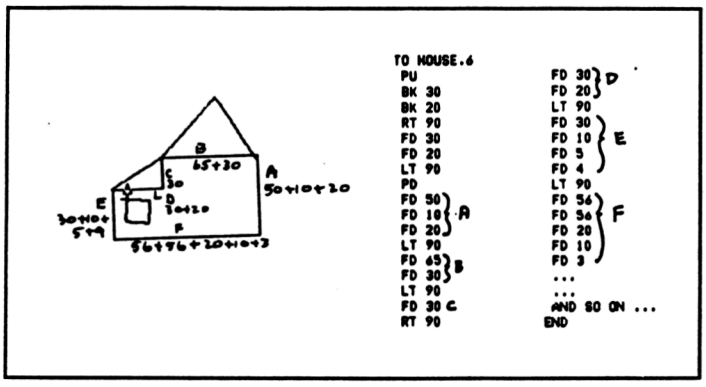


Figure 4.1. Kathy's HOUSE.6 (Grade 4)

Kathy's approach to Logo drawing is to try out a series of steps in direct mode and write down each step in her notebook as she goes along. When her picture looks the way she wants it to, she copies all the steps listed in her notebook and uses the Logo editor to add them to her procedure. In her HOUSE.6 project Kathy used visual problem solving (Kull, 1986) to decide how far to move the turtle. She used a combination of large and small steps until she got the turtle where she wanted it. Then she turned the turtle 90 degrees to go in another direction. If the FORWARD steps in Kathy's procedure are added up—a mathematical operation that Kathy almost certainly did not do—the sum reveals that her figure is slightly off. The total heights of the two sides do not quite match (see the lines labeled A, C, and E), although the lengths of the top and bottom do match (see the lines labeled B, D, and F).

Although visual problem solving is perfectly appropriate for many Logo tasks, Kathy did not use mathematical knowledge that might have been helpful to her. She did not add FORWARD steps together or use mathematical reasoning to determine that the two sides of a house should be exactly the same height. Nor did she calculate how far the turtle had turned. Kathy's window is askew because her method of solving problems visually failed her due to an inconsistency in Logo graphics on the Apple computer. Although the turtle in the figure *looks* as though it is heading straight up, its actual heading is slightly to the right of vertical. (The screen resolution and available memory for the Apple computer limits the turtle to 36 different positions as it rotates through 360 degrees. When the turtle is *nearly* vertical, it appears to be *exactly* vertical. Since Kathy relied on visual information to solve problems, she was totally baffled when her window was not precisely aligned with the horizontal and vertical.)

After examining printouts of this project at several stages, Kathy's teacher asked her to change her way of working: he suggested that after she got the turtle where she wanted it, she add up the FORWARD steps and use the total number of steps in her procedure. He also showed her how to use the HEADING command to determine the turtle's true heading. From that point on, Kathy's work became easier for her to carry out and easier for her teacher to understand when he looked at it! Chapter 10 presents a case study of how Kathy's teacher used his ongoing assessment of her work-in-progress to design teaching interventions that could support her in learning to use Logo more powerfully.

A. Estimating Angles and Distances

Developing the ability to visually estimate angles and distances on the screen is an important part of gaining control over the turtle. There seem to be significant individual differences, among adults as well as among children, in how quickly people learn to do this. Some children quickly develop the ability to estimate angles and distances as they learn to draw simple shapes with the turtle. Others may need to spend time with specific activities, such as a target game, in order to develop these skills (Watt, D., 1983). It is difficult to determine from the printout of a student's Logo project just how adept he or she may be at estimating. You have to actually observe students at work, infer estimation difficulties from the type of help they ask for, or ask them specifically whether estimation is difficult for them.

One subskill of estimation is understanding units: how big is 1 turtle step? 10 turtle steps? 100 turtle steps? How far does the turtle rotate when it turns RT 10, RT 90, and so forth? An understanding of 360 degrees and its factors (discussed in Chapter 3) is extremely helpful in estimating rotations. Another subskill is having a sense of order of magnitude: to move the turtle forward from point A to point B, will you need an input of 10, 50, 100, or 500 turtle steps? Figure 4.2, Vinh and Martha's Computer Project, shows an early stage of a project that eventually became much more elaborate. It illustrates how these students estimated sizes and combined their estimates with mathematical knowledge about the properties of rectangles to construct a well-proportioned figure made up of rectangular shapes.

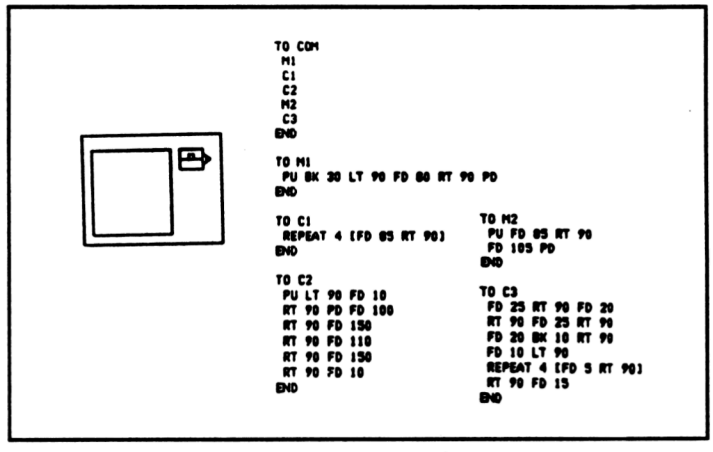


Figure 4.2. Vinh and Martha's Computer Project (Grade 5)

Procedure C1 draws the inner square (the monitor screen) with a size of 85. C2 draws the outline of the computer itself, using a rectangle with dimensions of 110 by 150. Notice how the last command of the C2 procedure, FD 10, completes the first side of the rectangle by adding 10 turtle steps to the FD 100 used to start drawing the rectangle. This indicates that Martha and Vinh knew that the opposite sides of a rectangle must be the same length—in this case, 110 turtle steps.

B. Combining Steps

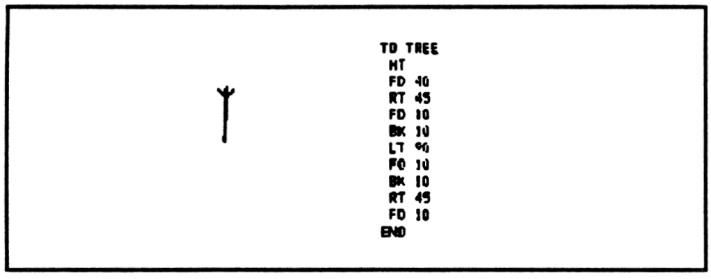
This is one of the simplest mathematical strategies Logo students use, although we can see in Figure 4.1 that Kathy did

not use it. A sequence of forward steps—FD 10, FD 20, FD 30—can be replaced by their sum, FD 60. Students can similarly combine a sequence of BACK, RIGHT, or LEFT commands. The difficulty comes when they have FORWARD and BACK, or RIGHT and LEFT, in the same sequence. Then they have to use the idea of inverses to subtract all the BACK steps from the FORWARD steps, or all the RIGHT turns from the LEFT turns, to get the correct total.

C. Using Inverses: RIGHT/LEFT, FORWARD/BACK

Students who recognize that FORWARD/BACK and RIGHT/LEFT can be used to undo each other or backtrack over a completed part of a drawing have learned a valuable problem-solving tool for turtle geometry, as well as an important mathematical abstraction—the concept of an inverse, a procedure or operation that reverses or undoes another (such as + and -, RIGHT and LEFT, FORWARD and BACK, PENUP and PENDOWN).

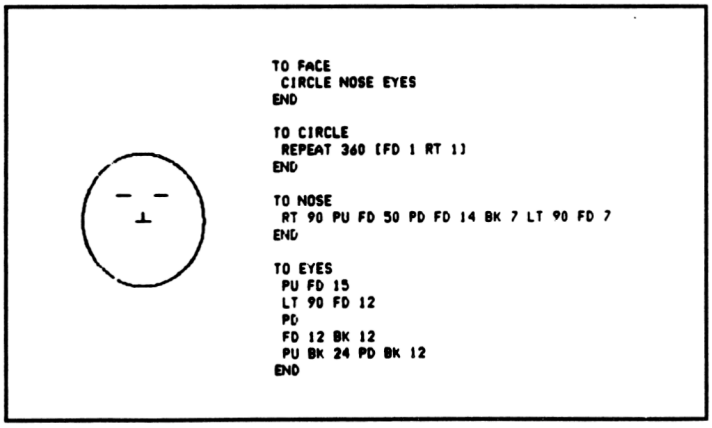
Figure 4.3, Jerry's TREE, which is part of a larger outdoor scene, offers a straightforward and fairly commonplace use of FORWARD/BACK and RIGHT/LEFT inverses to draw a symmetrical tree.



```
TO TREE
 HT
 FD 40
 RT 45
 FD 10
 BK 10
 LT 90
 FD 10
 BK 10
 RT 45
 FD 10
END
```

Figure 4.3. Jerry's TREE (Grade 4)

Figure 4.4, Sheila's FACE, provides another example of using FORWARD/BACK inverses and combining steps to create a symmetrical face.



```
TO FACE
 CIRCLE NOSE EYES
END

TO CIRCLE
 REPEAT 360 [FD 1 RT 1]
END

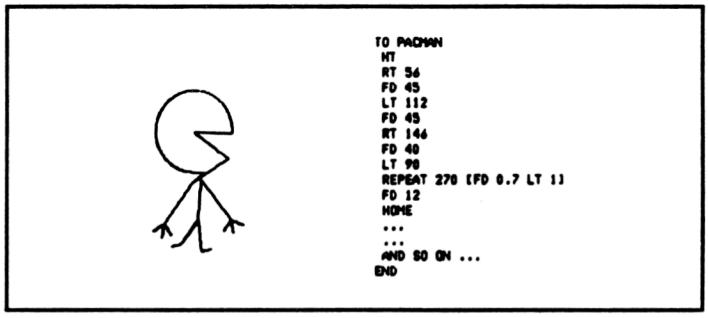
TO NOSE
 RT 90 PU FD 50 PD FD 14 BK 7 LT 90 FD 7
END

TO EYES
 PU FD 15
 LT 90 FD 12
 PD
 FD 12 BK 12
 PU BK 24 PD BK 12
END
```

Figure 4.4. Sheila's FACE (Grade 4)

The radius of Sheila's Logo circle is almost exactly 57 turtle steps. Sheila's NOSE procedure moved the turtle to the center and drew a centered line-segment, using steps of PU FD 50 PD FD 14 BK 7. We don't know whether Sheila used the turtle to measure the diameter of the circle experimentally before making the NOSE procedure, or whether she simply used a combination of estimation and experimentation until the picture looked right to her. Her EYES procedure also makes use of combining steps and FORWARD/BACK inverses to produce two eyes that are located symmetrically to the right and left of a vertical line passing through the center of the circle.

Douglas' PACMAN Design in Figure 4.5 shows a striking use of both RIGHT/LEFT inverses and combining steps to draw the head of his Pacman.



```
TO PACMAN
 HT
 RT 56
 FD 45
 LT 112
 FD 45
 RT 146
 FD 40
 LT 90
 REPEAT 270 [FD 0.7 LT 1]
 FD 12
 HOME
 ...
 ...
 AND SO ON ...
END
```

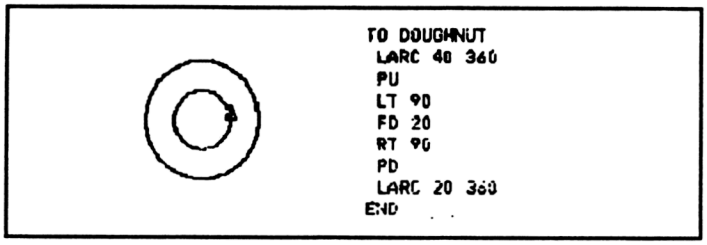
Figure 4.5. Douglas' PACMAN Design (Grade 4)

This figure includes just part of his procedure. Look at the numbers he used for his turtle rotations: RT 56, followed by LT 112, followed by RT 146. At first they might seem to be unrelated choices determined by estimation and experimentation. But if we notice that LT 112 is the same as LT 56 + LT 56, and that RT 146 is RT 56 + RT 90, we can infer that Douglas used RIGHT/LEFT inverses and combining steps to make his design come out exactly the way he wanted it to look. We don't know whether he calculated these numbers in his head or on paper or whether he used the uncombined numbers in his original design and combined steps later. Douglas' mastery of the 360-degree circle and the use of prevariable thinking are shown in the way he used REPEAT 270 [FD 0.7 LT 1] to draw a three-quarter circle of exactly the size he wanted. These are powerful examples of a student's mathematical thinking embedded in a Logo design, and of the kind of detective work a teacher might need to do to uncover it!

D. Using Specific Knowledge About Particular Shapes

A student who understands the special properties of a square, rectangle, triangle, or circle can use this understanding to solve problems much more quickly than someone who has to figure out each step visually. Vinh and Martha's Computer Project (Figure 4.2) shows a good use of the properties of squares and rectangles. In Sheila's FACE (Figure 4.4) and Douglas' PACMAN (Figure 4.5), both students use specialized knowledge about Logo circles to complete their designs.

Brad's DOUGHNUT, shown in Figure 4.6, uses tool procedures that draw circular arcs.



```
TO DOUGHNUT
 LARC 40 360
 PU
 LT 90
 FD 20
 RT 90
 PD
 LARC 20 360
END
```

Figure 4.6. Brad's DOUGHNUT (Grade 4)

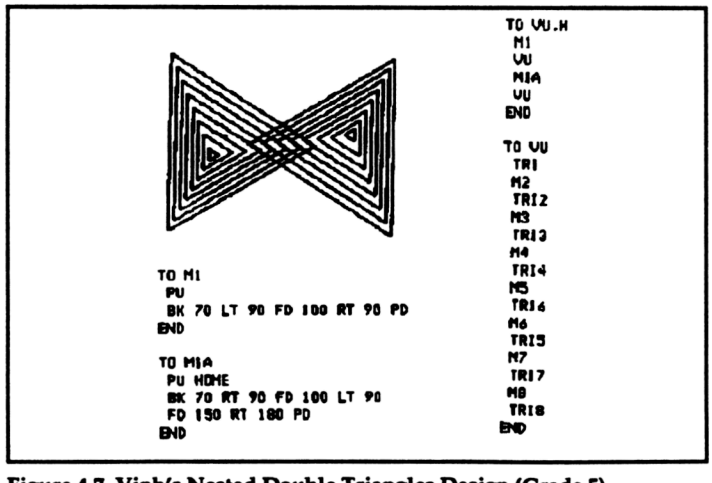
The first input to LARC determines the radius of the arc, while the second determines the total angular rotation. Brad used 360 as LARC's second input to draw a full circle. His DOUGHNUT procedure draws centered circles by starting with a circle of radius 40, moving 20 turtle steps toward the center, and drawing a second circle of radius 20. If Brad does not yet fully understand the properties of circles, this little project offers an opportunity for teaching him more about circles and radii.

E. Using RT/LT Symmetry

RIGHT/LEFT symmetry is a special kind of inverse relationship that is extremely useful in solving many Logo problems and simplifying others. To make one figure symmetrical to an existing figure, the student can simply reproduce all the commands in the first figure and replace each rotation by its inverse. Or, within a symmetrical figure, such as a tree or person, the student can make one side symmetrical to the other by using the same commands with inverse rotations. Jerry's TREE (Figure 4.3) is a nice example of this, while Sheila's FACE (Figure 4.4) uses FORWARD/BACK inverses to produce a figure with RIGHT/LEFT symmetry.

F. Using Similarity

To make the same shape in different sizes, the student can hold all the angles constant but change the sizes proportionally. This can be quite simple to do when working with regular figures such as squares or equilateral triangles. Vinh's Nested Double Triangles Design in Figure 4.7 demonstrates a very simple use of similarity: all equilateral triangles are proportional.



```
TO VU.H
 M1
 VU
 M1A
 VU
END

TO VU
 TRI
 M2
 TRI2
 M3
 TRI3
 M4
 TRI4
 M5
 TRI5
 M6
 TRI6
 M7
 TRI7
 M8
 TRI8
END

TO M1
 PU
 BK 70 LT 90 FD 100 RT 90 PD
END

TO M1A
 PU HOME
 BK 70 RT 90 FD 100 LT 90
 FD 150 RT 180 PD
END
```

Figure 4.7. Vinh's Nested Double Triangles Design (Grade 5)

The side lengths used in procedures TRI-TRI8 vary from 150 to 10, with decreases of 20 between successive triangles. To create similar shapes with more complex figures, proportional variables are required. Diane's SAILBOAT, shown in Figure 2.6, is an example of this.

G. Using Transformational Geometry

This involves figuring out how to redraw a given shape in a new position, rotate it into a new orientation, change its size, or flip it symmetrically about a given line. These techniques are useful in mastering turtle graphics, and they are also an important part of many elementary and middle school math curriculums. (This is often more true in Canada than in the United States.)

Vinh's Nested Double Triangles in Figure 4.7 shows another approach to RIGHT/LEFT symmetry using transformational geometry. His work is interesting for its use of slides and rotations to position figures. Each of the internal move procedures, M2-M8, uses exactly the same sequence (including RIGHT/LEFT inverses) to move the turtle from one triangle to the next. The procedures M1 and M1A that move the turtle before drawing each set of triangles demonstrate the use of transformational geometry ideas (slide-turn-slide-turn and so on), along with RIGHT/LEFT symmetry. M1A moves the turtle home, then reverses the angles used in M1 to move the turtle to the right side of the screen. Then, because the VU procedure has all triangles turning to the right, Vinh had to move the turtle 150 steps forward and reverse its direction to begin the design with the turtle in the upper right corner.

Although Vinh's mathematical thinking seems rather sophisticated, both in Figure 4.7 and in Figure 4.2, he clearly has a lot to learn about using procedures effectively in a modular fashion (see Chapter 1) and is probably ready to learn to use variables as well (see Chapter 2).

H. Using and Understanding Wrapping

Screen wrapping is one of the most fascinating and least understood aspects of turtle geometry, partly because it has almost no analogs in noncomputer geometries. The very first Logo turtles were mechanical robots that moved and drew pictures with real pens on a floor or table. The distance a floor turtle can move is constrained by the length of its control wires or the distance over which its control signals are transmitted. But theoretically, if you command a floor turtle to move FORWARD 1000, it will keep going straight ahead until it travels a distance of 1000 turtle steps.

When a simulated turtle was first programmed for a video screen, moving the turtle for long distances was prohibited. A FORWARD command that took the turtle beyond the edge of the video screen resulted in an "out of bounds" error message. One benefit of this restriction was that it forced Logo programmers to engage in significant mathematical problem solving to ensure that their turtle designs would stay on the screen. They had to consider the scale of a turtle design and make it small enough and position it correctly so that no command would result in an error message.

A drawback to this was that it made Logo more difficult for novices, and especially for young children. An alternative was established: a Wrap command was introduced that would allow the turtle to move "beyond" the edge of the screen in a rather peculiar way. When the turtle moves forward to a position that is beyond the boundaries of the original screen, it does not stop or cause an error message. Instead, it moves instantly and invisibly to the opposite boundary of the screen and keeps going from there, as shown in Figure 4.8, Illustration of Turtle Wrapping.

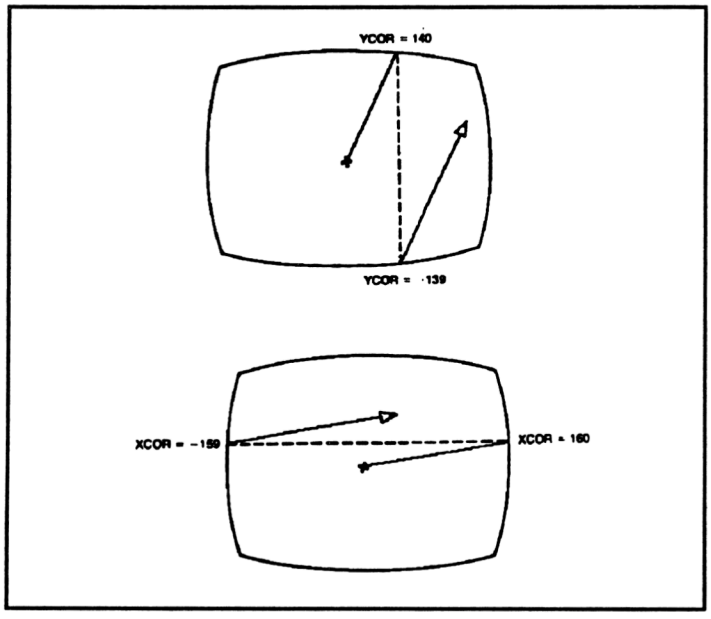


Figure 4.8. Illustration of Turtle Wrapping. (Reprinted, with permission of Addison-Wesley Publishing Company, from *Teaching with Logo: Building Blocks for Learning*, by Watt and Watt, © 1986.)

Most versions of Logo now come with wrapping as the default mode for turtle behavior. Some allow alternate modes: a NOWRAP or FENCE mode that produces the "out of bounds" error message and a WINDOW mode in which the turtle disappears without an error message if it is sent off-screen, from whence it can reappear if it is given commands that will bring it back within the original screen boundaries. However, some versions of Logo allow only the wrapping mode with no alternatives.

Wrapping extends the turtle geometry microworld in delightful and seemingly mysterious ways. Although the rules governing wrapping are just as mathematically predictable as those governing other turtle commands, its effects are difficult for inexperienced Logo users to understand. The invisibility of the wrapping movement and the lack of a physical parallel—it's virtually impossible to simulate wrapping effectively by playing turtle—combine to make wrapping a mysterious, rather than an easily understood, phenomenon.

Although many students may not understand wrapping, they love to explore its effects. One reason is that fascinating designs can be created with little apparent thought or effort. Teachers, on the other hand, often discourage students from extended explorations with wrapping. This may be due in part to the fact that children seem to be creating effects without understanding them mathematically and therefore may be "wasting time" that could be used more productively, for example, by exploring phenomena that they can more easily control. It may also be partly because the teachers themselves do not understand the mathematics of wrapping or how to use wrapping to stimulate students' thinking and problem solving.

Despite the almost universal appeal of wrapping phenomena to students, there has been almost no research on what students learn when they engage in wrapping. Seymour Papert (1980) does not mention wrapping in *Mindstorms*, and there are few curriculum materials or Logo books that discuss wrapping in any depth (see Watt and Watt, 1986b, for an exception). It seems to be the case that most students who explore wrapping do so without much specific assistance from a teacher (and sometimes against a teacher's instructions to the contrary). This could make the ways that students use and understand wrapping a wonderful topic for research about independent mathematical problem solving in a complex and unpredictable environment. Yet, most Logo researchers, including the teachers in our collaborative research project, have chosen to study other aspects of learning rather than to concentrate on wrapping.

One approach we have used to help students understand wrapping is to introduce tool procedures that slow down the turtle and make the effects of wrapping more visible (Watt & Watt, 1986a). A consequence is that when a student uses large inputs, the process of wrapping is clearly indicated on the screen. The slower process also takes a long time, which forces a student to deliberately choose smaller inputs to produce specific effects rather than wait a long time to see the effects of large, unplanned inputs.

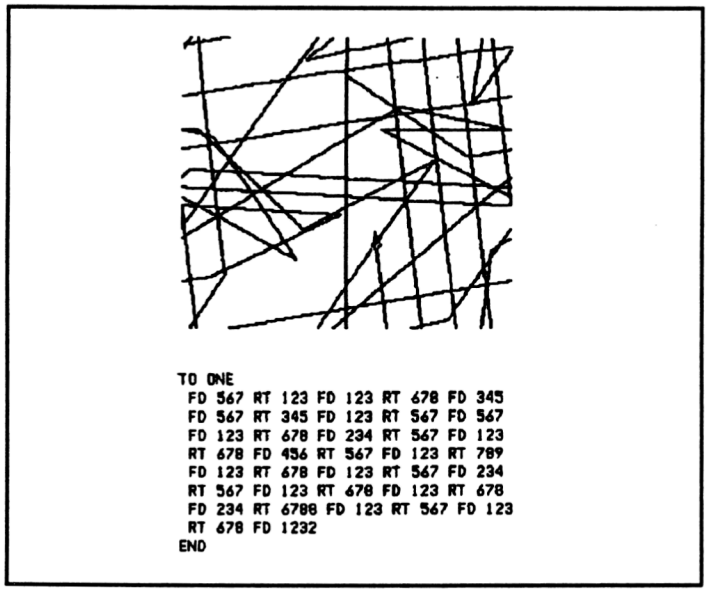
Although teachers in our collaborative research project have not gathered a lot of data about students' work with wrapping effects, we can illustrate one student's progress toward control of wrapping. Explorations with wrapping phenomena often start with more or less random commands involving large inputs. Brad's Work in Figure 4.9 shows how one student progressed from apparently random wrapping effects to carefully planned, predictable effects without assistance from a teacher.

Brad's inputs to both FORWARD and RIGHT are clearly influenced more by number sequences (123, 567) than by any thought about what effects a particular input might have. Because the consequences of large inputs are unpredictable at first, students may not distinguish between the effects of large distance inputs, which do cause significant wrapping effects, and large angle inputs, which do not. (Recall that an angle input larger than 360 has the same effect as the same input minus 360.)

Another stage in Brad's development in working with wrapping is illustrated in Figure 4.10, Brad's AS Procedure.

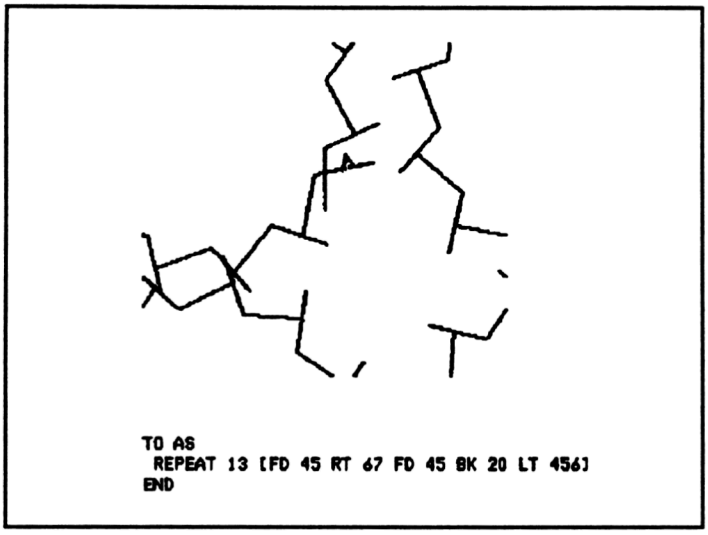
Although the result of Brad's new procedure, AS, is still largely unpredictable to him, both its visual design and its command structure are much easier to understand. Inputs to FORWARD, BACK, RIGHT, and LEFT are still primarily number sequences (45, 20, 67, 456), and Brad still does not

seem to have sorted out the effects of distances and rotations. However, the use of REPEAT makes it possible to achieve large-scale effects with relatively short command lines.



```
TO ONE
 FD 567 RT 123 FD 123 RT 678 FD 345
 FD 567 RT 345 FD 123 RT 567 FD 567
 FD 123 RT 678 FD 234 RT 567 FD 123
 RT 678 FD 456 RT 567 FD 123 RT 789
 FD 123 RT 678 FD 123 RT 567 FD 234
 RT 567 FD 123 RT 678 FD 123 RT 678
 FD 234 RT 6788 FD 123 RT 567 FD 123
 RT 678 FD 1232
END
```

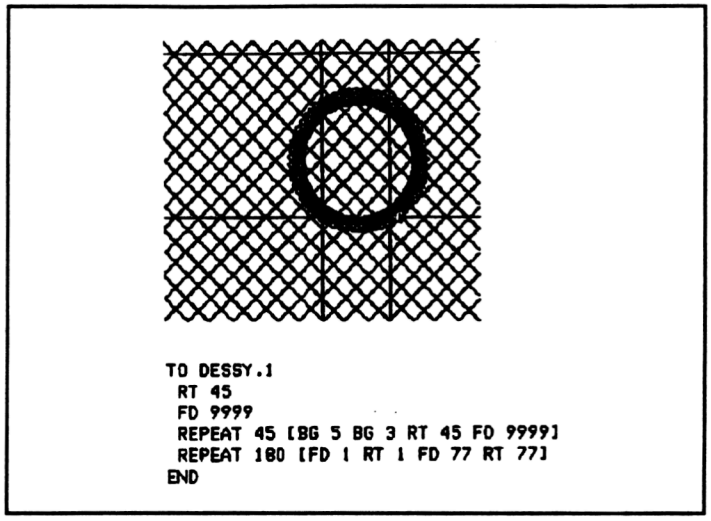
Figure 4.9. Brad's Work (Grade 4)



```
TO AS
 REPEAT 13 [FD 45 RT 67 FD 45 BK 20 LT 456]
END
```

Figure 4.10. Brad's AS Procedure (Grade 4)

In Brad's DESSY.1 Design, shown in Figure 4.11, Brad seems to be much more in control of the effects of wrapping. In this project, Brad clearly knows that the wrapping effect is caused by large inputs to FORWARD and that 45-degree angles produce a simple, predictable pattern on the screen. He is also in enough control of his effects that he can use wrapping to produce a patterned background with a separate, wreath-like design in the foreground. However, his choice of 45 for the number of REPEATs shows that he is not fully aware of how many REPEATs are needed to produce the effect he is trying to create.



```
TO DESSY.1
 RT 45
 FD 9999
 REPEAT 45 [BG 5 BG 3 RT 45 FD 9999]
 REPEAT 180 [FD 1 RT 1 FD 77 RT 77]
END
```

Figure 4.11. Brad's DESSY.1 Design (Grade 4)

Finally, Brad's D.5.4 procedure shows him in full control.

```
TO D.5.4
  RT 1
  REPEAT 8 [FD 9999]
  BG 1 BG 2 BG 3 BG 4 BG 5 BG 6
END
```

Here, Brad uses the smallest angle he knows about and the smallest number of REPEATs needed to fill in the entire screen. Brad's work shows that it is possible for students to make progress in their understanding and control of the mathematics of wrapping—in this case without a significant amount of teacher involvement. It also shows us some possible paths that teachers could follow in designing challenges to support students learning in this provocative yet confusing mathematical environment. For instance, students could be asked to focus on the rotation used before wrapping (Brad used RT 1), on the number of REPEATs (Brad used 8), or on the size of the FORWARD steps (Brad used 9999). Slowing down the turtle (as described in Watt and Watt, 1986a) can also be useful in helping students understand wrapping.

A Closing Thought

Some Logo research studies have shown that students may not fully understand the mathematical and programming ideas they have actually used in their work (see, for example, Pea, Kurland, and Hawkins, 1987). Indeed, the students who carried out the projects described here may not have recognized that they were using mathematical thinking. One responsibility of a Logo teacher is to teach appropriate mathematical strategies to students who are not using them (Clements, 1990). Another responsibility is to detect the mathematical inventions students are already using and help them become more aware of the general principles and strategies involved (Watt & Watt, 1986b). If Logo teachers can encourage students to understand, describe, and share the mathematical ideas they are already using, we'll be well on the way to creating the kind of mathematical culture Seymour Papert had in mind.

Chapter 5
Strategies for Learning Through Exploration

Levels of Critical Aspect 5

A. Keeping Records of Exploratory Work

B. Taking Advantage of Accidents

C. Using and Looking for Patterns

D. Testing the Limits

E. Making and Testing Hunches and Hypotheses

F. Generalizing From One or More Examples

G. Sharing Ideas With Others

Is Brad Wasting His Time?

Brad is a fourth grader who engages in lots of Logo explorations. According to his teacher,

> he works at an average fourth-grade level in all areas ... although he needs a lot of teacher reminding, both behaviorally and academically.... However, when it comes to working with Logo, Brad has remarkable perseverance. He is anxious to complete tasks he sets for himself and appreciates teacher help when stumped.

Nevertheless, his teacher was concerned about what he was learning and was often uncertain about how to help him. For one thing, he had a great many apparently unrelated procedures in his workspace. For another, he had lots of different files on his disk, all of which contained many procedures, that were only slightly different.

Some of Brad's procedures were part of a large city project he had worked on for some time. Others seemed to be unrelated to anything in particular. Many of them used REPEAT, often with large numbers. There were many changes of pen color and background color. When he was asked what the different procedures did or why he had made them, Brad had difficulty remembering.

What was Brad learning from his explorations? Was he wasting his time, or did his work have some purpose he could not articulate and his teacher couldn't easily detect? In the previous chapter, we looked at some examples that demonstrated that Brad had been making progress toward understanding and controlling the phenomenon of screen wrapping. But how could his teacher support him in using exploratory learning more effectively and in being more aware of what he was learning through his explorations? These are the types of questions we address in this chapter.

Exploratory Learning in a Logo Microworld

Logo was designed to be a microworld for learning mathematics, computer science, and problem solving. A microworld is a specifically constructed, carefully restricted learning environment in which intriguing phenomena are connected to powerful ideas (Lawler, 1982; Watt, D., 1985b). Papert (1980) described microworlds as "incubators for knowledge" (p. 120). His intention was to create environments in which learning occurs in a personal and playful fashion as the learner uses ideas embedded in the microworld to invent a game, draw a cartoon, command a robot, or make a movie. Robert Lawler (1982) has explained that "microworlds are in essence 'task domains' or 'problem spaces' designed for...streamlined experience.... The appropriation of the knowledge...is made possible because the microworld does not focus on 'problems' to be done but on 'neat phenomena'—phenomena that are inherently interesting to observe and interact with" (p. 140).

One major challenge for Logo teachers is to support students in learning through their own explorations rather than merely solving problems or meeting challenges the teacher sets. In this chapter, we address some of the learning strategies students can use to further their own learning while engaging in open-ended explorations.

Process Knowledge Versus Content Knowledge

In this chapter and the following two chapters, we focus on some of the *process* components of Logo learning rather than on the *content*. In previous chapters we shared examples of how students have used some of the specific content embedded in Logo by its developers: programming ideas involving procedures and variables and mathematical ideas embedded in turtle geometry. Our source of knowledge was clear and specific—we looked at actual printed examples of student's work, either completed or in progress.

In looking at the *processes* by which students actually construct knowledge while working on their Logo projects, we must confess that we're on much shakier ground than when looking at the *content* of their learning. Examples of students' printed work can tell us a great deal about which ideas they are using and how their ideas have changed over time, but they give very few hints as to how they selected the particular ideas they decided to use, how they moved from idea to idea, how they collaborated with other students, and how they handled problems they encountered along the way.

In our collaborative research project (Watt, D., 1989; Watt & Watt, 1988), each teacher selects two students to observe as often as possible, regularly collects their work in progress (including their written plans, notes, and hand drawings), and engages these students in conversations about what they are doing and thinking. One teacher in the project used IBM Logo dribble files—exact records of every command her students actually typed (Olive, 1986). Most of the teachers who work with us report that insights gained by carefully following two students helped them support the learning of many of the other children in their classes as well. (See Chapter 10 for a case study that illustrates this.)

Classroom teachers who want to understand more about how their own students work through projects and gain knowledge might also benefit by reading reports of careful research by other observers. An outstanding book, *Cultivating Minds*, by Sylvia Weir (1987), offers a number of mini-case-studies of Logo learning-in-process. The *Final Report of the Brookline Logo Project* also offers a number of detailed case studies of Logo students at work (Papert et al., 1979; Watt, D., 1979a, 1979b). Readers interested in more formal research can refer to Douglas Clements' (1988-89) series of columns on Logo and problem solving that appeared in *Logo Exchange*.

A. Keeping Records of Exploratory Work

Logo students who regularly record what they have done in the form of handwritten notes, sketches, and printouts of Logo pictures and procedures are much better able to recall what they have done, why they did it, and what order they did things in. They can look back to see how they did something before and therefore don't have to endlessly repeat previous work. However, experience has shown that Logo students rarely keep thorough records on their own. Teachers can support record keeping in many ways. They can teach students to print out their own work, have them keep regular journals, and have them keep records of exploratory projects on forms designed specifically for that project.

If dribble files are available, students can use them to trace their own work. For example, Andy first created the design shown in Figure 5.1, Andy's Design, by typing TRIANGLE, LT 5 over and over again, using REPEAT and individually typed commands. When his work session was over, he asked his teacher for a copy of his dribble file. He then added up all the times that the sequence TRIANGLE, LT 5 was repeated and wrote the procedure LOTS.

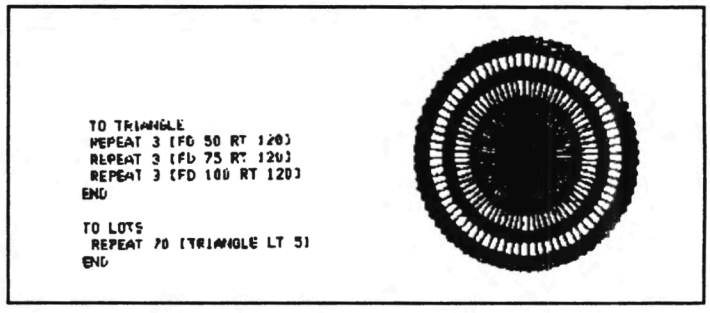


Figure 5.1. Andy's Design (Grade 5)

B. Taking Advantage of Accidents

Explorations can lead to useful results that can be used for purposes other than what a student might have originally had in mind. For example, one day Brad experimented with rotating figures and produced the work shown in Figure 5.2, Brad's Rotated Designs.

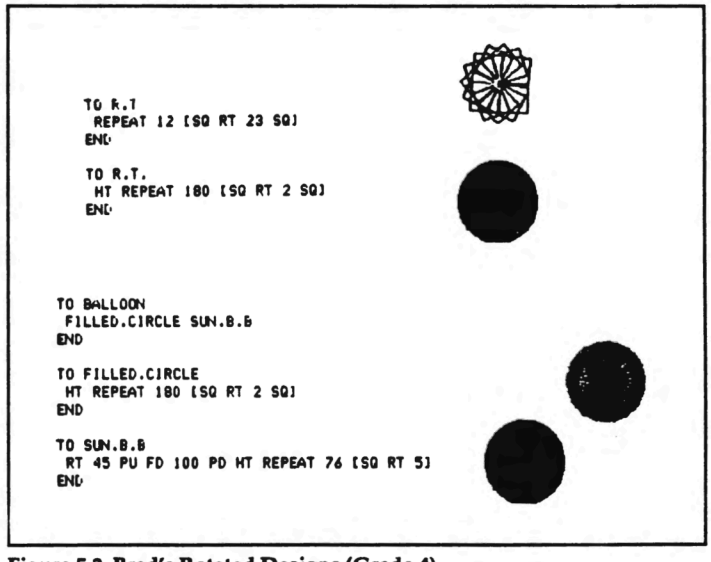


Figure 5.2. Brad's Rotated Designs (Grade 4)

In attempting to complete the rotated design R.T, Brad used many different values for the number of REPEATs and the angle of rotation. One of them created a filled-in circle to which he gave a new name, R.T. (that is, he added a period to

the T on the earlier name, R.T). Brad, who was extremely interested in filling shapes, used this idea in a later project, BALLOON. (As you can see, he was also exploring the effects of different backgrounds and pen colors.)

C. Using and Looking for Patterns

Andy was interested in complex geometric shapes. He experimented with nested shapes and rotated shapes, as shown in Figure 5.1 and in the illustration shown in Figure 5.3, Andy's TRIANGLE and SPIRALS Procedures.

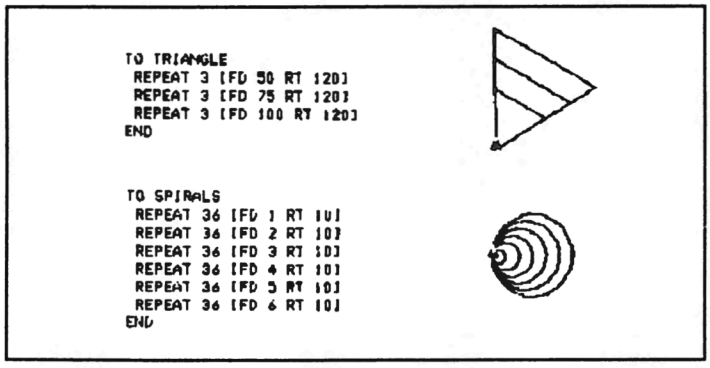


Figure 5.3. Andy's TRIANGLE and SPIRALS Procedures (Grade 5)

Notice how Andy used a pattern—a constant increase in size—to draw both his TRIANGLE and SPIRALS procedures. As shown in Figure 5.4, Andy's PENTAGON and HEXAGON Procedures, he attempted to use a similar pattern to go from a pentagon to a hexagon: increase the number of REPEATs by 1, and both the size and angle by 10.

Although Andy was looking for patterns and regularities in his work, he had not yet discovered the significance of 360 degrees for drawing regular shapes (see Chapter 3).

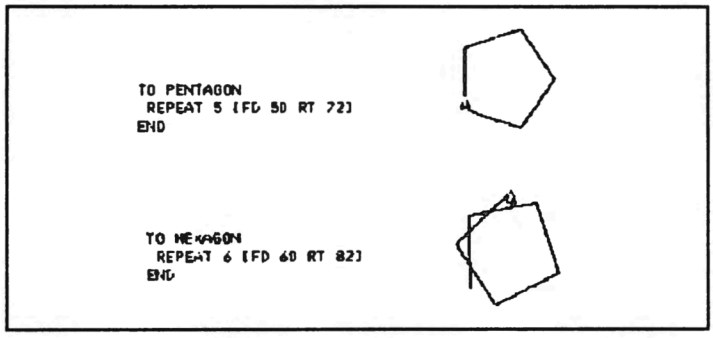


Figure 5.4. Andy's PENTAGON and HEXAGON Procedures (Grade 5)

D. Testing the Limits

Understanding extreme cases, that is, using large and small numbers, is often a very useful exploration strategy. Some of Brad's explorations are shown in Figure 5.5 (Brad's CHEEZE Design), Figure 5.6 (Brad's DESSY.1 Design), and Figure 5.7 (Brad's D.5.4 Procedure).

In his CHEEZE design, Brad uses 9999 as a standard "large number" and does not really know the connection between the number of REPEATs, 50, and the angle through which the figure is rotated (rotating the turtle LT 9999 is equivalent to rotating it LT 279).

In his DESSY.1 procedure, Brad uses FD 9999 to produce wrapping effects, while the angle, 45 degrees, is chosen specifically to produce the effect shown. The number of repeats, however, is far larger than needed for the design.

Brad's Procedure D.5.4 uses a large distance, FD 9999, and the smallest angle Brad knew about, RT 1, to produce a very specific intentional effect; and the number of REPEATSs is exactly the number needed to fill the screen.

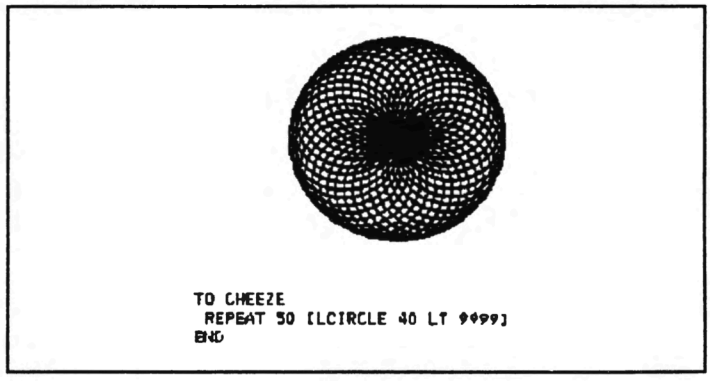


Figure 5.5. Brad's CHEEZE Design (Grade 4)

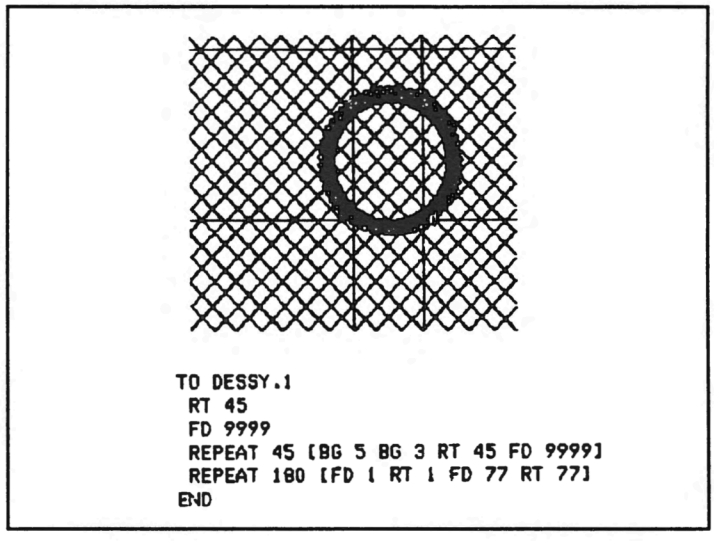


Figure 5.6. Brad's DESSY.1 Design

```
TO D.5.4
  RT 1
  REPEAT 8 [ FD 9999]
  BG 1 BG 2 BG 3 BG 4 BG 5 BG 6
END
```

Figure 5.7. Brad's D.5.4 Procedure (Grade 4)

E. Making and Testing Hunches and Hypotheses

Students should be encouraged to guess, to predict, and to formulate rules as part of the process of exploration. They should also be encouraged to write their rules down, describe how they tested them, and report on their results.

Andy kept records of his work and could tell his teacher exactly which ideas he was testing (see Sections A and C of this chapter). However, many students move back and forth between random exploration and careful testing of hunches or looking for patterns. If they do not keep records, they may forget about the thoughtful problem solving they were doing and explain that they were "just messing around." Brad was a frequent victim of his own amnesia in this way. His teacher was able to conclude that his work was valuable only by carefully analyzing it after the fact!

F. Generalizing From One or More Examples

Brad's work at filling in squares, shown in Figure 5.8, Brad's CITY.2 Project, provides an excellent example of how an exploration can be adapted, modified, and eventually generalized. Brad explained that his first fill-in procedure was R.T. (see Figure 5.2). As he observed how a circular shape was filled in by taking small rotations, he got an idea for filling in a square shape by drawing a series of lines one turtle step apart.

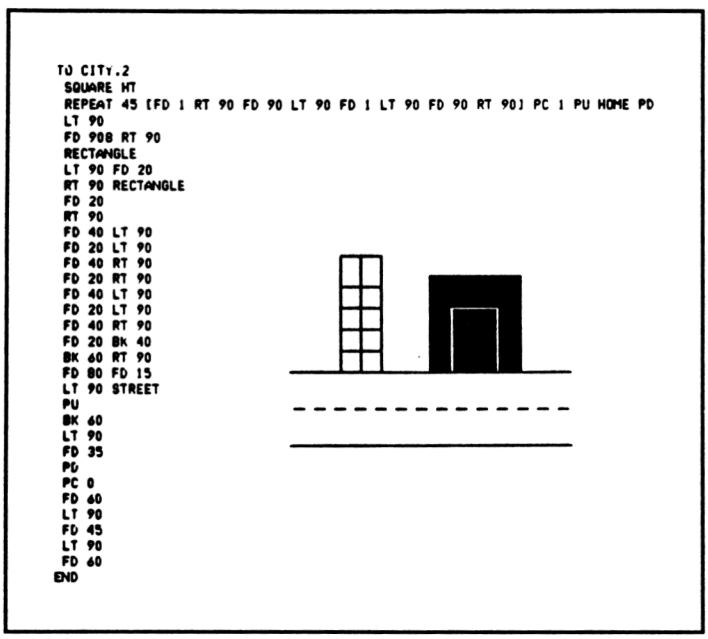


Figure 5.8. Brad's CITY.2 Project (Grade 4)

Brad used the command

```
REPEAT 45 [ FD 1 RT 90 FD 90 LT 90 FD 1 LT
90 FD 90 RT 90]
```

to fill in a 90 x 90 black square and draw a garage for his CITY.2 project.

Next he used the same technique with different numbers to fill a 45 x 60 rectangle. Brad's OPEN Procedure, shown in Figure 5.9, used with the turtle's pen set to the background color, draws an open door.

Later he used the same approach to fill a 30 x 30 square to represent a truck, as shown in Figure 5.10, Brad's TRUCK Procedure.

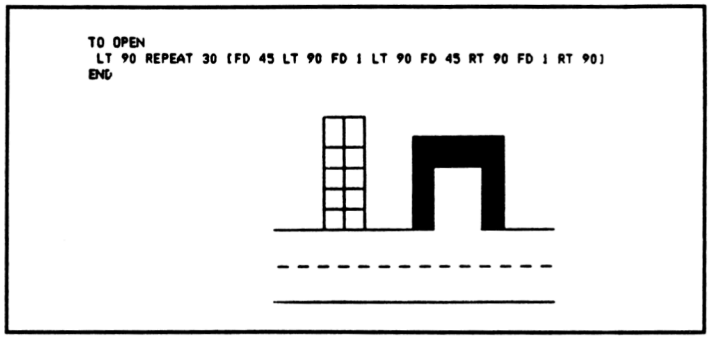


Figure 5.9. Brad's OPEN Procedure (Grade 4)

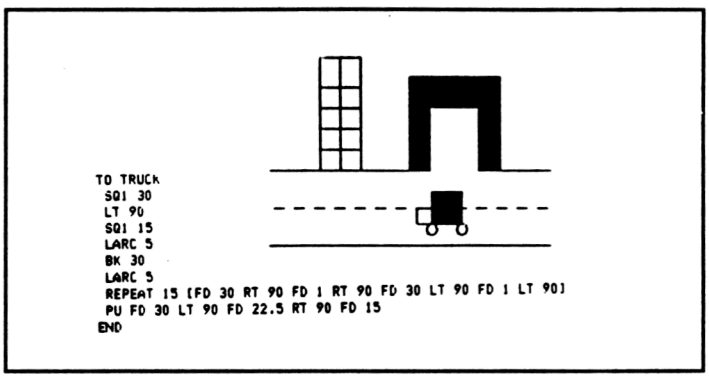


Figure 5.10. Brad's TRUCK Procedure (Grade 4).

Brad's ultimate objective was to have the garage door open, the truck come out, and then have the door close. Brad's CLOSE Procedure, shown in Figure 5.11, is similar to OPEN but is used with the turtle's pen set to the original color. CLOSE was used to refill the empty doorway so that it appeared to close.

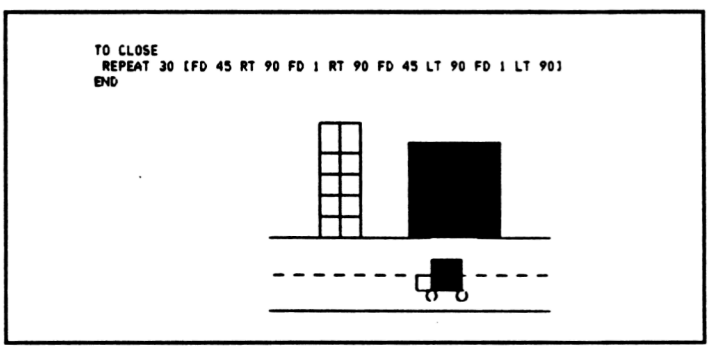


Figure 5.11. Brad's CLOSE Procedure (Grade 4)

Eventually, Brad's teacher helped him generalize his method for filling squares by showing him how he could write a procedure that used a variable to fill in any square:

```
TO FILL.SQ1 :SIDE
  SQ1 :SIDE
  REPEAT :SIDE/2
    [FD :SIDE RT 90 FD 1 RT 90
    FD :SIDE LT 90 FD 1 LT 90]
END
```

The ability to generalize from examples—to see how an idea that someone invents or discovers in one context can be used in other contexts—is at the very heart of learning through exploration. Teacher intervention is essential for this to occur, because the student's attention is focused only on the immediate problem at hand. Brad took one step toward generalization by himself—he changed a command used to fill in a 90 x 90 square so that he could fill in a 30 x 30 square. His teacher showed him how to fill *any* square by using variables, and by doing so provided both a reusable tool and a model for generalizing a Logo procedure. Without this intervention, generalization might not occur. Many such interventions might be needed before students can articulate for themselves the power that comes from generalizing.

G. Sharing Ideas With Others

Finally, we want to emphasize the importance of learning from others as a key strategy for learning through exploration. Once Brad figured out how to fill in a square, several of his classmates began to experiment with filled-in shapes themselves. Brad's persistence in filling in shapes and his willingness to share his work with others led to many fruitful explorations on the part of his classmates. A classroom climate that encourages sharing of work in process, of the process itself, and of finished work, is a critical factor in successful exploratory learning.

Chapter 6
Debugging Strategies

Levels of Critical Aspect 6
Debugging Strategies

A. Identifying the Bug

B. Locating the Bug

C. Fixing the Bug

D. Incorporating the Bug

E. Reflecting on Bugs to Help Eliminate Future Bugs

F. Using Planning Strategies to Help Eliminate Potential Bugs

The Trouble With Kathy's House

In the first part of Chapter 4, we looked at Kathy's HOUSE.6 project (Figure 4.1) as an example of visual problem solving. We'll now focus on one detail of her project, the bug in the turtle's orientation that led to what Kathy, a fourth grader, called a "lopsided" window, and we'll see how her teacher helped her debug it. Figure 6.1, Kathy's House, shows her original House project with the "lopsided" window.

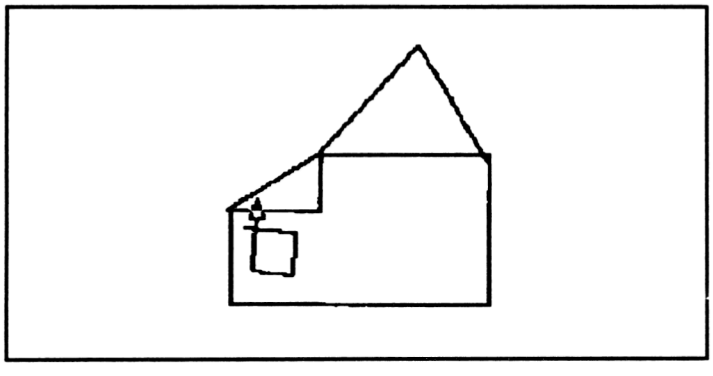


Figure 6.1. Kathy's House

As can be seen from the figure, the turtle looked—to Kathy and to us—as if it was heading straight up the screen. However, the lines it drew when it moved forward were at a slight angle. The problem occurred because the net result of a series of rotations, which Kathy very carefully constructed through a process of successive approximations, left the turtle at a heading of 95 degrees when it looked as though it was at a heading of 90 degrees, just before starting to draw the window. This effect is shown in Figure 6.2, Kathy's Turtle Rotations.

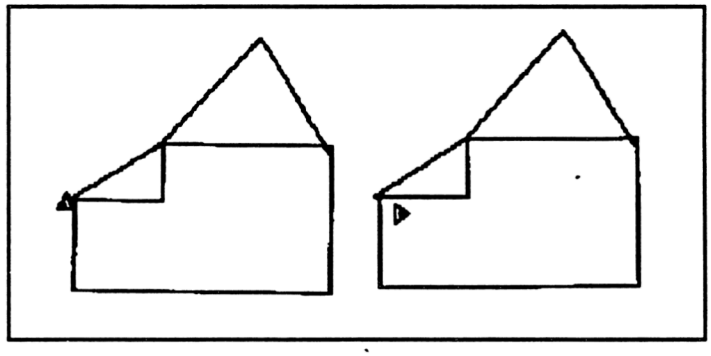


Figure 6.2. Kathy's Turtle Rotations

Kathy's approach to drawing Logo pictures relied almost entirely on being able to determine the turtle's position and orientation visually. When its visual image was slightly out of alignment with its actual orientation, she was stumped. Kathy's teacher showed her how to use the TRACE and PRINT HEADING commands and suggested that she make the turtle's heading zero before starting to draw the window. Kathy then found that the turtle's heading was 239 degrees just after it finished drawing the roof. By trial and error she found that a rotation of RT 121 made the turtle's heading zero. She erased all the commands in her procedure that came after drawing the roof, added RT 121, and completed the rest of her house successfully, using her tried-and-true method of visual problem solving. Her final results are shown in Figure 6.3, Kathy's Completed House.

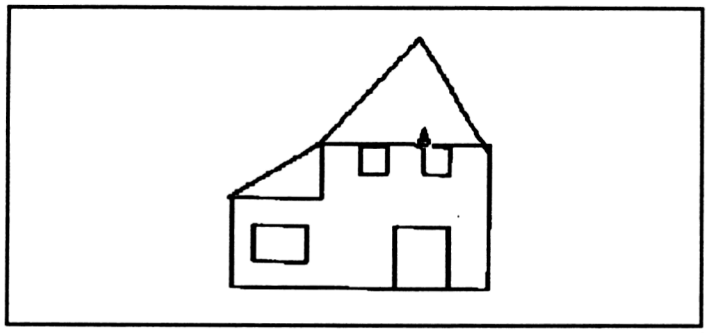


Figure 6.3. Kathy's Completed House (Grade 4)

Debugging did not come naturally to Kathy. Her teacher had to encourage her to investigate the causes of her problem and show her a specific strategy she could use. The ways that Kathy's teacher helped her become a more powerful Logo user are described more fully in Chapter 9.

The Central Importance of Debugging

If the impact of computers in classrooms were limited to one idea and we were allowed to choose what that idea could be, we would choose the idea of a bug—a mistake or misconception in a computer program—and the associated notion of debugging. The Piagetian view of learning, on which Logo's educational philosophy is based, suggests that learning is a process of building internal models of aspects of reality and refining those models by testing them, gradually revising them when they no longer seem to fit our real-world experiences. This means that the most important types of learning occur through a process of trial and error, especially through a process of compensating for errors by changing the models guiding our actions. This implies that the errors we make, and our own thinking that leads to those errors, are as important to study as the actual tasks we are trying to accomplish.

Seymour Papert (1980) made debugging one of the central themes of *Mindstorms* and contrasted it with the right-or-wrong ethic that we encounter so often in classrooms (especially in math class!):

> School teaches that errors are bad; the last thing one wants to do is pore over them, dwell on them, or think about them.... The debugging philosophy suggests an opposite attitude. Errors benefit us, because they lead us to study what has happened, to understand what went wrong, and through understanding, to fix it. Experience with computer programming leads children more effectively than any other activity to "believe in" debugging. (p. 114)

Yet even in Logo classrooms we see learners (adults as well as children) attempt to hide and ignore their bugs, clear away the evidence, and start all over without attempting to understand what has just happened. Papert attributed this phenomenon to the cumulative effects of schooling:

> The ethic of school has rubbed off too well. What we see as a good program with a small bug, the child sees as "wrong," "bad," "a mistake."... The child is glad to take advantage of the computer's ability to erase it all without any trace for anyone to see.... Contact with the Logo environment gradually undermines long-standing resistance to debugging. (p. 114)

We agree that difficulties with debugging are partly a matter of attitude. But we also think that there are specific skills and strategies that make debugging more effective and encourage people to engage in debugging more often, which in turn helps them shift their attitudes about making errors. Some of these are quite simple and straightforward. Others involve making use of special capabilities of particular versions of Logo. Still others involve taking the time to take note of and think about bugs as interesting, in and of themselves. We'll discuss debugging and describe some key debugging strategies in this chapter.

The Joy of Debugging

Jim MacIsaac was the most experienced Logo programmer in a class of 27 classroom teachers taught by Molly Watt in Vancouver, Canada, a few years ago. Most of them were learning Logo for the first time, and Jim spent a great deal of time helping his colleagues through their programming difficulties—so much so that several of them were feeling guilty about taking him away from his own work. Sensitive to these feelings, he surprised the class one day by spontaneously thanking them for providing him with such a wonderful problem-solving experience. "The part of Logo I love best is debugging," he explained. "But since I've been programming in Logo for a while now, I have great difficulty coming up with interesting bugs on my own. In this class you've provided me with so many intriguing bugs to work on, really challenging problems to solve, many more than I could possibly have invented in months of trying. It's been lots of fun, and a terrific learning experience for me."

It should be a goal of all Logo teachers to encourage the Jim MacIsaacs in our Logo classes, and to use them as models to help all our students learn to enjoy the process of debugging as one of the most essential, interesting, and pleasurable parts of any learning experience.

A General Debugging Strategy

David Klahr and Sharon Carver, cognitive scientists working at Carnegie Mellon University, have developed a general approach to debugging Logo programs and have reported positive learning gains (that is, better debugging and more success in completing projects) as a result of teaching a specific debugging curriculum to children as one part of their Logo learning experience (Carver, 1988; Carver & Klahr, 1986). In its most simplified form, their approach consists of three distinct steps: first, identify the bug; next, locate the bug (that is, find out exactly where the bug occurs in a particular set of procedures and subprocedures); and,

finally, fix the bug. We'd also like to suggest that there are two other aspects of debugging worthy of the attention of Logo teachers and students: (1) Incorporate the bug, that is, take advantage of a new possibility that came about because your procedure did something you had not intended; and (2) develop new programming strategies that make the possibility of repeating the same bug much less likely in the future.

A. Identifying the Bug

A.1. Making use of Logo's error messages. Although this seems obvious, it is often neglected. How often have you seen Logo students press the Return key several times to clear the screen and remove the evidence of error in the form of a Logo error message? Logo's error messages are designed to help a user identify exactly why a command or procedure could not be carried out and where the problem occurred. Students should learn to consider an error message as a source of useful information. This is a first step in learning to identify, locate, and fix those bugs that occur as a result of typing commands that Logo cannot carry out.

Teachers can help by asking students to pay attention to error messages and to ask themselves what caused the message and what they need to do to avoid similar messages in the future.

A.2. Describing the bug. This is exactly what Kathy did when she told her teacher that she was unhappy with the window in her house because it was "lopsided." It was what Douglas did when he wrote "I wish the design would go in each part of the circle" on a print of his pentacircle design shown in Figure 6.4, Douglas' PENTACIRCLE.

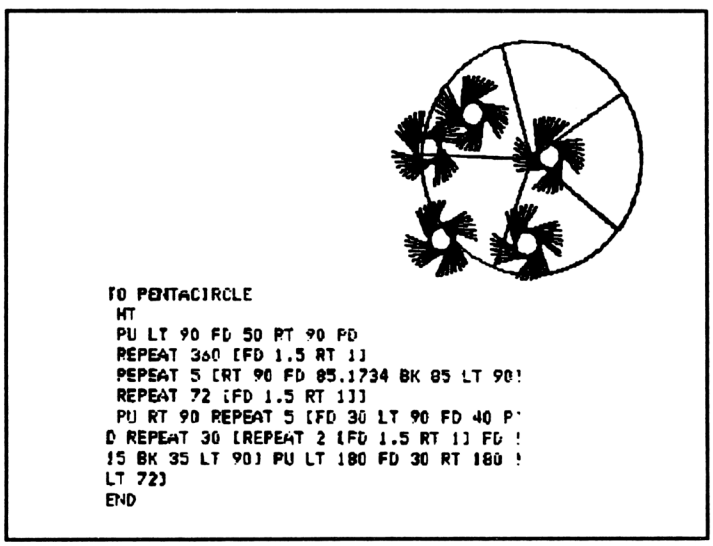


Figure 6.4. Douglas' PENTACIRCLE (Grade 4)

In our experience with hundreds of Logo students of all ages, describing a bug to a teacher or classmate is often all that is needed for a programmer to say "Oh, now I see what happened," and be well on the way to resolving the problem (Watt & Watt, 1986b).

A.3. Categorizing and naming the bug. This can be extremely helpful in debugging future instances of the same or similar bugs. Both Kathy's and Douglas' bugs could be called "turtle-state bugs," and even that general descriptor helps us see where to start debugging. More specifically, Kathy's bug might be called: "The turtle looks like its heading is 0 or 90 degrees, but its actual heading is slightly off." If Kathy had this description written down on a list of common turtle-state bugs, she might have known what to do about it on her own.

A.4. Recognizing familiar bugs. Naming and describing Douglas' bug in Figure 6.4 is a little more complicated. However, we actually think it's a combination of two rather common turtle-state bugs: "Failure to distinguish between the part of a procedure that moves the turtle and the part that draws the design" and "failure to have the turtle return to its exact starting place, in drawing a distinct part of a design, before moving to draw another part of the design." In addition, we'd say that Douglas' procedure exhibits two other common bugs in Logo problem-solving strategies: "Hiding the turtle to make it draw faster before having completed a project" and "failure to subdivide a complex project into clearly named subprocedures." We believe that if Douglas had been aware of these four descriptions of common types of bugs, he would have been much less stumped by the bug in his procedure.

B. Locating the Bug

B.1. Using Logo debugging aids. Let's not forget the obvious. Logo provides lots of built-in debugging help: error messages, TRACE. SHOWTURTLE, HEADING, PRINTOUT (PO), and so on. If the bug involves a command that Logo could not carry out, Logo's error messages tell what that command is, and in some versions of Logo it identifies the procedure and even the line in which the command occurred. Following error messages, a TRACE primitive (in some versions of Logo) is Logo's most powerful debugging aid. TRACE lets a procedure do its work, one line at a time, so that a user can isolate exactly where something went wrong. For a turtle-state bug, using TRACE in conjunction with PRINT HEADING may give you all the information you need to figure out what went wrong, as well as where. Since only a few command lines can be shown on the screen at once, a printout on paper is another crucial debugging aid. (Believe it or not, we've encountered many Logo students and even Logo teachers who have never learned to make a printed copy of their pictures or their procedures.)

B.2. Dividing a long procedure into parts. Sequential, undifferentiated Logo procedures are notoriously difficult to debug, but even a long procedure can be divided into parts if you learn to look for certain clues. The most common are the use of PENUP and PENDOWN in many turtle geometry designs to separate the parts of the drawing that move the turtle from the part that draws the design. Mark the location of PENUPs and PENDOWNs on your printed copy, and work from there. For designs in different pen colors, commands that set the pen color can be used the same way to differentiate parts of the design and isolate the bug.

A procedure is easier to debug if it's divided into parts, each of which carries out only one task, such as moving the

turtle or drawing a particular piece of a design. If a procedure is too complex to understand easily, such as Douglas' PENTACIRCLE, rewriting the procedure by dividing it into subprocedures and testing each subprocedure separately might be an important step in locating the bug.

C. Fixing the Bug

Sometimes all you need to do is locate the bug, and you'll see exactly what needs to be done to fix it. In other cases, such as Kathy's House project, there is still more work to be done.

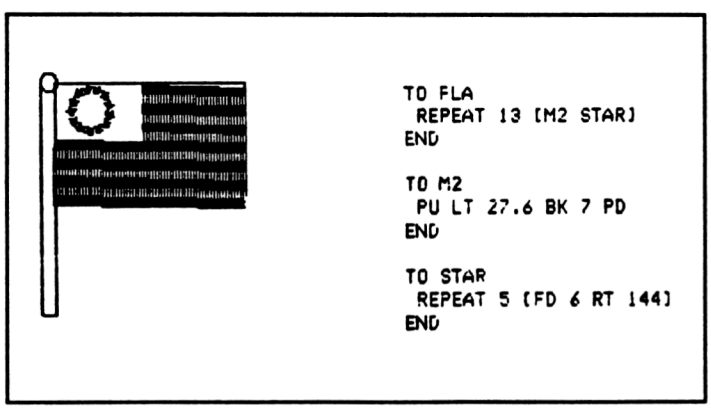
C.1. Playing turtle. If your bug is a turtle-state bug, playing turtle often helps unsnarl what went wrong and helps you see what to do about it.

C.2. Using mathematical knowledge. Mathematical knowledge, such as knowing the Rule of 360 (total turtle trip), can be invaluable in correcting bugs. If you add up how far the turtle has already turned, you may be able to determine the correct amount to turn next by using the idea that a complete turtle rotation is 360 degrees, or that a symmetrical figure requires a left turn in a certain place when a right turn was used in a corresponding place somewhere else. This approach might be very helpful to Douglas. He seems to use the total turtle trip idea whenever he can. However, in his PENTCIRCLE procedure shown in Figure 6.4, he has not applied it correctly to the very long last line, which begins with PU RT 90 and includes a complex nested REPEAT command.

C.3. Testing each part separately. Some combination of this approach and the preceding ones will probably be essential to fix Douglas' bug. Unless he identifies, by name, the part of his procedure that draws each of the little whirligig designs and the part that moves it from place to place, he could use trial and error forever, with little hope of solving his problem.

D. Incorporating the Bug

Sometimes a bug improves a project because it introduces an element you might never have planned. Martha and Vinh's FLA Procedure, shown in Figure 6.5, is a case in point.



```
TO FLA
 REPEAT 13 [M2 STAR]
END

TO M2
 PU LT 27.6 BK 7 PD
END

TO STAR
 REPEAT 5 [FD 6 RT 144]
END
```

Figure 6.5. Martha and Vinh's FLA Procedure (Grade 5)

After Martha and Vinh completed this procedure entirely on their own, their teacher asked them two questions: "How did you figure out the spacing of the thirteen stars?" and "How did you get the stripes to ripple?" Neither the teacher nor her students realized that there was a direct connection between the two processes (see Section E of this chapter). This took a careful study of the printout and some Logo detective work.

Martha and Vinh explained that they had divided 360 by 13 and used the result, 27.6, to rotate the turtle between stars. (This is clearly shown in three of their subprocedures, FLA, M2, and STAR.) They went on to explain that the ripple of the stripes was an accident! They had expected the stripes to come out straight, but when they came out at a slight angle, they left them that way because it produced a much better looking flag than the one they had planned.

Incorporating a bug can sometimes be a cop-out, an excuse for being lazy. And frankly, unless a student has a good reason for fixing a program, such as publishing it for other students to use, there may be little justification for requiring students to fix every bug unless they are dissatisfied with their results. Therefore, creating a context in which debugging is appreciated, in which bugs themselves are valued and shared as much as final results, is a critical element of teaching effective debugging strategies.

E. Reflecting on Bugs to Eliminate Future Bugs

Martha and Vinh never debugged their flag, and neither did their teacher. But even when you have incorporated a serendipitous bug into your final design, it's useful to play detective and see if you can determine how it came about. Experienced turtle geometers will no doubt recognize that the two students were victims of the same bug as the one in Kathy's House project—the turtle looks like its heading is 0 or 90 degrees, but its actual heading is slightly off. But where did the bug come from? We need to call on the total turtle trip idea and use PRINT HEADING to complete the detective work. Try this. Start with the turtle at home and type REPEAT 13 [RT 27.6] PRINT HEADING. You'll find that the turtle's heading is closer to 359 degrees than 360. This makes the turtle draw a jagged line rather than a straight one. Thereby hangs the ripple!

What happens after a bug is eliminated or incorporated? Remember Piaget's idea about learning? We only learn by confronting discrepancies between our existing model of how something works and our experience of what really happens when that model is applied. After debugging and resolving an immediate difficulty, it may be useful—perhaps essential—in maximizing the benefits of learning Logo that we stop and reflect to see if we can change the way we do things so that in the future we'll be less likely to be plagued with the same bug all over again. Sometimes the reflection is initiated by a student, sometimes by a teacher. For instance, Kathy's teacher insisted that Kathy use subprocedures with separately named MOVE procedures in her next Logo project! Douglas' teacher suggested a similar strategy to him. Martha and Vinh's teacher had already established the use of subprocedures in her students' work. They were less likely to get stumped by bugs they wanted to fix than were Douglas or Kathy.

F. Using Planning Strategies to Help Eliminate Potential Bugs

Although the topic of this chapter is debugging, we'd like to close by suggesting a few strategies for minimizing bugs—at least of the turtle-state variety:

- Take careful notes, draw pictures of what the turtle is doing, write notes on the pictures, use dotted lines to show where the pen is up, and so on.
- Use subprocedures. Name each part of a turtle drawing and name the moves and picture elements separately.
- Wherever possible, make use of mathematical ideas, such as the total turtle trip, special angles, RIGHT/LEFT symmetry, and the characteristics of special shapes.
- Create state-transparent designs in which the turtle returns to its starting place after drawing each part before it moves to draw something else. If Douglas had done this, he'd have been home free!

Is There Life After Debugging?

We're certainly not suggesting that Logo teachers strive for bug-free programming by their students. First, it's impossible! Second, it would spoil lots of the fun (remember Jim MacIsaac)! Finally, if we managed to avoid all bugs, we'd have stopped learning because we'd only be working with what we already knew!

Chapter 7
Planning, Carrying Out, and Completing a Logo Project

Levels of Critical Aspect 7
Planning, Carrying Out, and Completing a Logo Project

A. Setting Appropriate Goals

B. Creating a Working Draft

C. Revising and/or Elaborating on a Plan

D. Sharing or Publishing the Results

Supporting a Logo Programming Process

One of the fundamental principles of Logo's educational philosophy is that students will learn mathematics, computer science, and problem solving by engaging in projects of their own choice and working through them in their own ways—with support from peers and teachers. We find this last point crucial. We have come to believe that support for all phases of project development, from the initial choice of the idea to final publication and sharing of the results, is an essential ingredient of an effective Logo learning environment.

Over the past few years we have developed a working description of what we call a "Logo programming process." We have observed that the development of a typical Logo project goes through several distinct phases: brainstorming and exploration; choosing a project; planning and simplifying the project; creating a working draft; debugging, revising, and elaborating; and sharing through publication (Thyberg, 1987; Watt & Watt, 1987a, 1987b, 1989). Our description is influenced by descriptions of the writing process articulated by Donald Graves (1983) and others, and used by thousands of writing teachers all over the world.

Figure 7.1 shows two schematic diagrams of the Logo Programming Process.

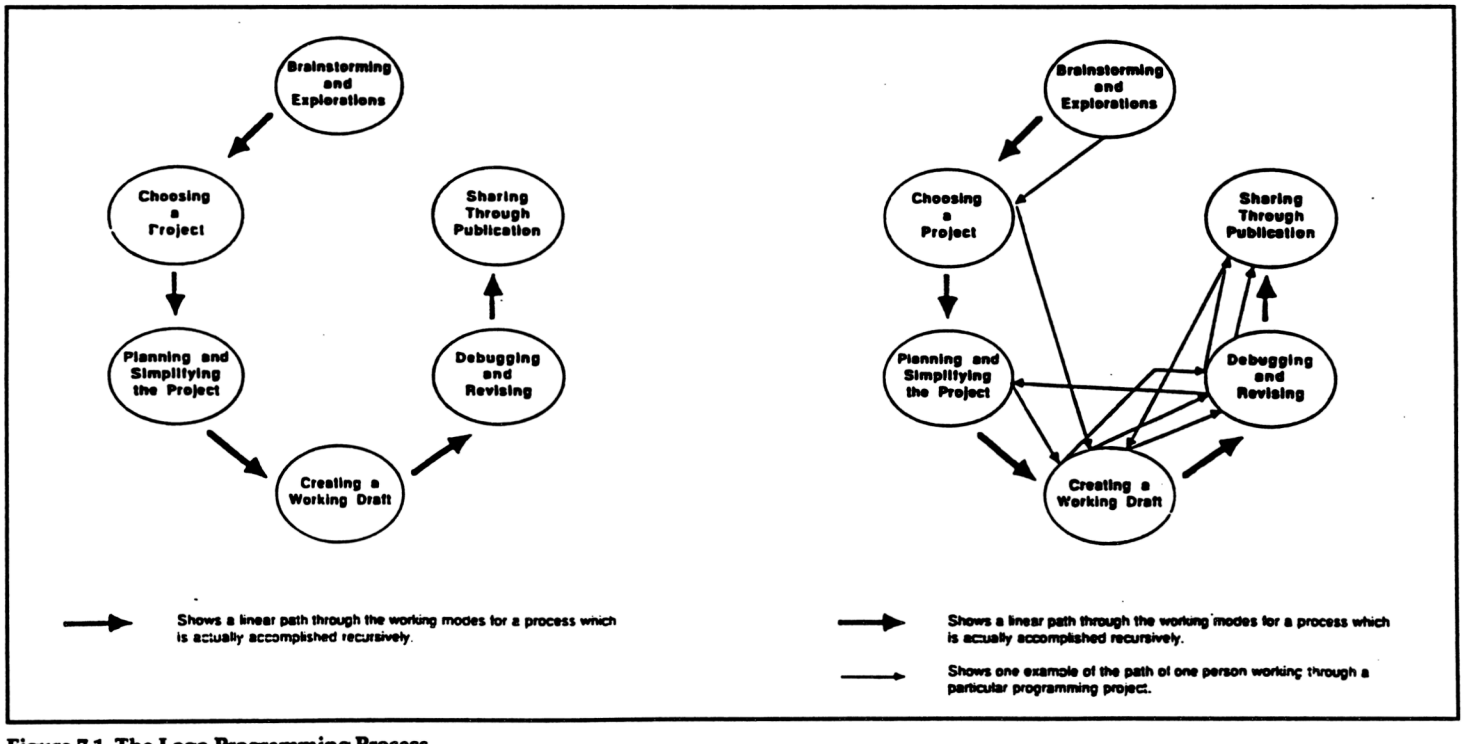


Figure 7.1. The Logo Programming Process

Notice that the process one goes through in completing a project is not the "ideal" linear one shown at the left in Figure 7.1. Instead, as shown in the more realistic diagram on the right in Figure 7.1, the process is recursive, moving back and forth between different working modes as the project develops over time.

The Statue of Liberty Project

In this chapter we will show how two sixth graders, Heather and Joanie, progressed through several phases of project development to carry out one long-term project—drawing the Statue of Liberty. By the spring of sixth grade, the girls had worked with Logo for almost three years. They were extremely comfortable with turtle graphics, with using variables to draw shapes such as rectangles, with dividing complex projects into small subprocedures, and with a structured planning approach their teacher had emphasized. Our research data about their project includes their drawings; written plans for procedures; printouts of computer procedures, including some that were discarded; notes taken by an observer (a computer coordinator in their school district); and a copy of a published article they wrote about their project for a class newsletter.

One extremely interesting feature of this particular project is that half-way through it, the girls discarded their original plan, threw out all the programs they had written, and completed the project successfully using an entirely different approach.

A. Setting Appropriate Goals

A.1. Using pictures and sketches as planning aids. A.2. Dividing a problem into parts. A.3. Using a superprocedure hierarchy and/or procedure tree in planning. Heather and Joanie's teacher insists that his students engage in a careful planning process before starting to type their programs on the computer. Figure 7.2, Heather and Joanie's Statue of Liberty Sketch, shows the elements that they planned to include in their program.

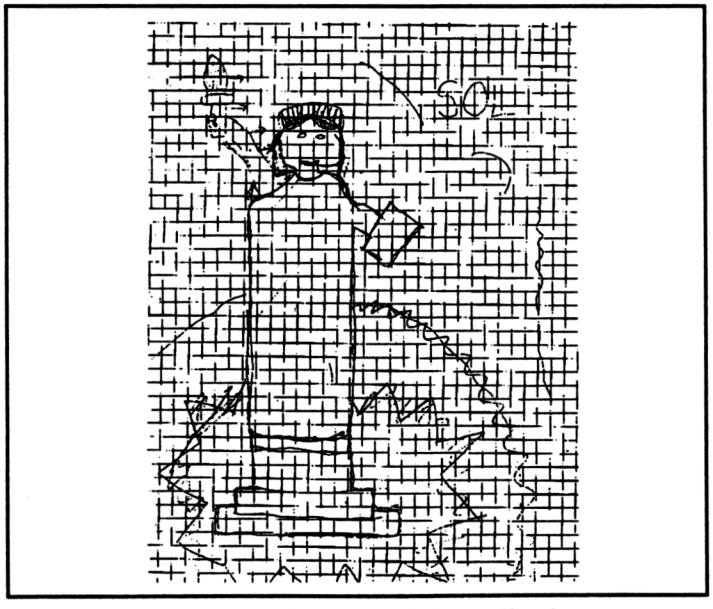


Figure 7.2. Heather and Joanie's Statue of Liberty Sketch

Figure 7.3, Heather and Joanie's Hierachy Chart, shows the chart (which we call a procedure tree) that they developed for the superprocedure and subprocedures they anticipated writing to draw the entire Statue of Liberty.

B. Creating a Working Draft

B.1. Simplifying the plan. B.2. Setting specific preliminary and intermediate goals. Heather and Joanie continued to work away from the computer. Figure 7.4, Heather and

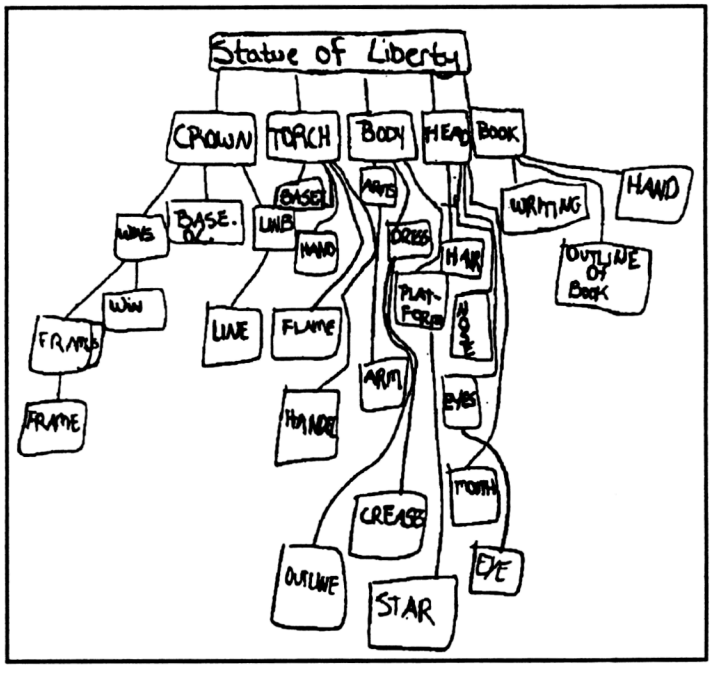


Figure 7.3. Heather and Joanie's Hierarchy Chart

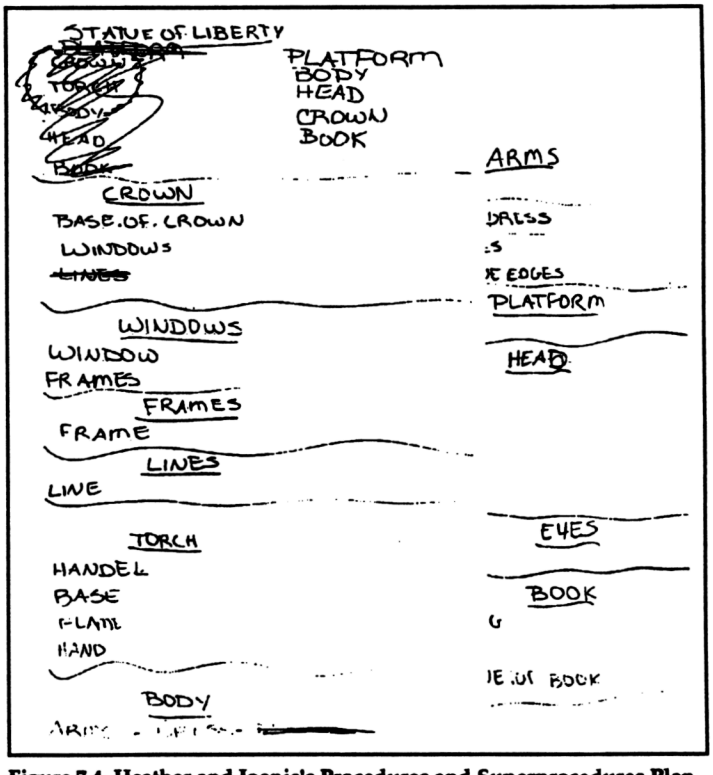


Figure 7.4. Heather and Joanie's Procedures and Superprocedures Plan

Joanie's Procedures and Superprocedures Plan, shows an elaboration of their hierarchy chart into specific procedures and subprocedures.

The girls' markings at the top of Figure 7.4 show that they had changed their minds about the best way to actually construct the statue. Instead of starting with the crown and working their way down to the base, they decided it would be easier to start from the bottom of the screen and work their way up. At this point they identified some preliminary goals. First they planned to draw the outline of the entire figure, next the crown, then the torch, and finally the book. After they completed all these steps, they planned to make the figure more elaborate by putting in all of the details shown in their sketch.

B.3. Beginning to write and test procedures. Heather and Joanie began working out and typing the procedures and subprocedures they needed. They began with a superprocedure they called SECRET, and they also wrote out all the main subprocedures as smaller superprocedures before actually filling in any details. Figure 7.5, Heather and Joanie's Statue Project—Intermediate Stage, shows their project before its completion.

All the major subprocedures are written as shells, with the sub-subprocedures still undefined. Only the platform and body subprocedures have been substantially completed. (The girls expressed some concerns that the project was more difficult to complete than they had expected.)

C. Revising and/or Elaborating on a Plan

C.1. Eliminating goals that are too difficult. C.2. Adding new goals or revising the entire plan as a result of new ideas. At this point, something unexpected happened. Heather and Joanie's teacher taught a lesson to the entire class, showing how to plot points on the screen using SETXY. He also taught his students to use a subprocedure, TO S, which allowed them to separately adjust the X and Y coordinates of a completed figure:

```
TO S :X :Y
  SETXY (:X *17) (:Y * 14)
END
```

Students can change the numbers by which the X and Y inputs are multiplied to stretch their final drawings in the horizontal and vertical directions. Heather and Joanie chose the values 17 and 14 for their Statue of Liberty project after a good deal of experimentation.

To Heather and Joanie, this new technique seemed made-to-order for the complex shape they were attempting to draw with the turtle. They literally abandoned all their existing procedures and subprocedures and plunged into their new approach. Heather and Joanie's New Statue Sketch is shown in 7.6. It demonstrates how they started planning their drawing all over again, tracing a drawing of the Statue of Liberty from a class magazine and transferring it to graph paper.

```
TO SECRET
 PLATFORM
 BODY
 HEAD
 CROWN
 TORCH
 BOOK
END

TO PLATFORM
 HT PU HOME BK 119 RT 90 FD 50 LT 180 PD
 VRECT 100 10
 PU FD 10 RT 90 FD 10 LT 90 PD
 VRECT 80 10
 RT 90 FD 10 LT 90 FD 10 PD VRECT 60 20
 VRECT 60 10
END

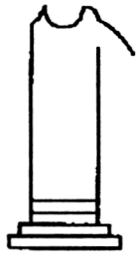
TO VRECT :W :L
 REPEAT 2 [FD :W RT 90 FD :L RT 90]
END

TO BODY
 DRESS
 ARMS
END

TO DRESS
 HT RT 90 PU FD 20 PD FD 130 BK 130 LT 90 FD 60 RT 90 FD 150 RT 55 REPEAT 20 [FD 1 LT 2]
 SETH 90 SETH 180 REPEAT 19 [FD 3 LT 10] SETH 90 REPEAT 20 [FD 1 RT 4]
END

TO ARMS
 SETH 120
 REPEAT 15 [FD 3 RT 2]
END

TO HEAD
 HAIR
 NOSE
 MOUTH
 EYES
END
```



```
TO CROWN
 BASE.OF.CROWN
 WINDOWS
END

TO TORCH
 HANDEL
 BASE
 FLAME
 HAND
END

TO BOOK
 WRITING
 HAND
 OUTLINE
END
```

Figure 7.5. Heather and Joanie's Statue Project—Intermediate Stage

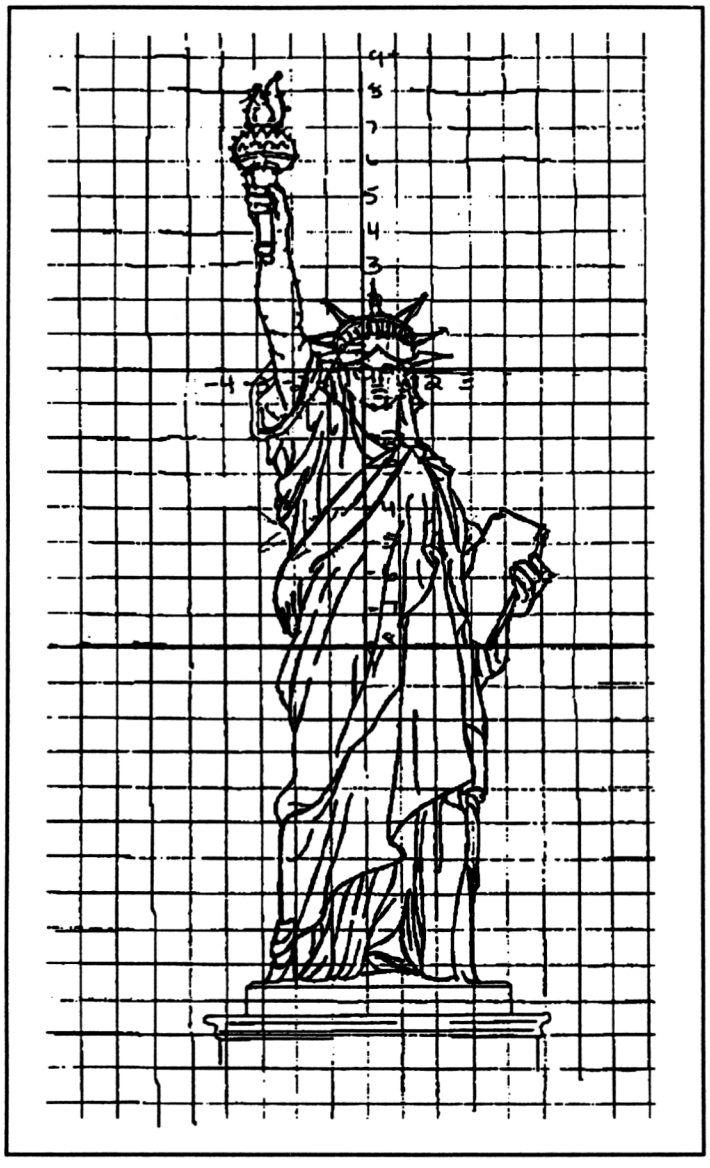


Figure 7.6. Heather and Joanie's New Statue Sketch

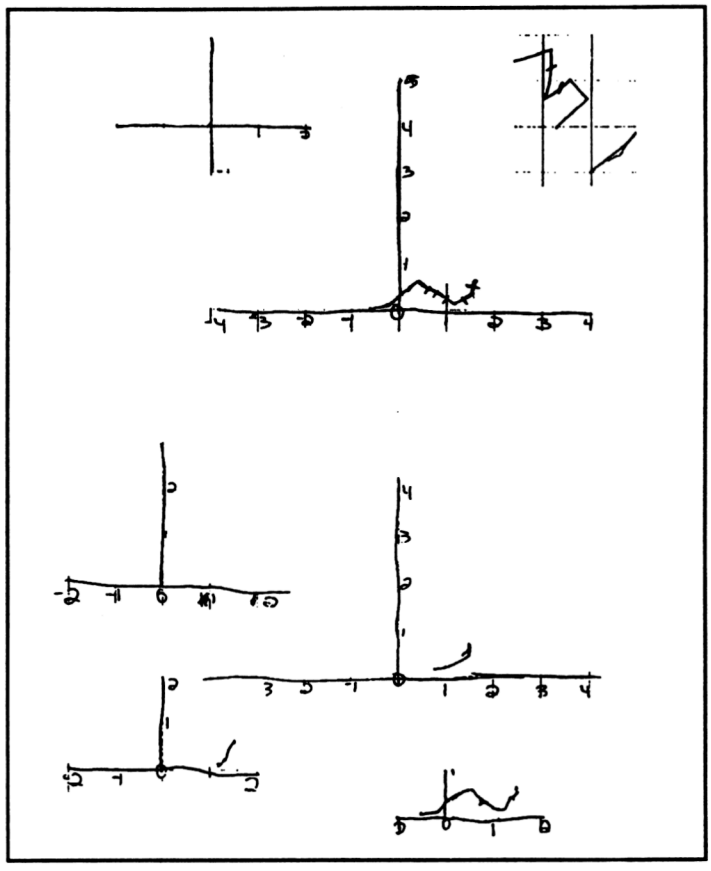


Figure 7.7. Heather and Joanie's Worksheet

Figure 7.7, Heather and Joanie's Worksheet, reproduces some of the work the two girls did as they continued to develop their project.

Heather and Joanie worked out the coordinates for every element in great detail, interpolating between grid lines to estimate coordinates to the nearest tenth of a unit. Finally they wrote each procedure by hand before typing it at the computer.

D. Completing the Project and Publishing the Results

D.1. Debugging the final version. As the girls later wrote in summarizing the process, "This was a difficult and very tedious task, but extremely rewarding. After we completed it, we did a lot of debugging." They were excited by the results, however, as each bit of the Statue of Liberty emerged from procedures that were, in themselves, virtually unreadable.

D.2. Elaborating on the final version. Heather and Joanie's final result is shown in Figure 7.8.

Although their work was still incomplete, they were quite satisfied with what they had accomplished.

D.3. Sharing the work in a public form. Heather and Joanie's teacher required that each major Logo project be written up in a newsletter format as a way of publishing the results for other students and parents. This gave Heather and Joanie an opportunity to explain their approach in written form. The results, which were produced in printed form using *Newsroom* software, are shown in Figure 7.9.

An interesting feature of their published results springs from something their teacher had taught them. By using the procedure TO S :X :Y (see Figure 7.8), they not only could draw the statue in proportion but they also could distort it and even reflect part of it about an axis by varying the scale factor used to multiply the X and Y coordinates used in TO S.

We encourage Logo teachers to help their students find an audience for their Logo work. Publishing Logo work in a newsletter, book, or even on a bulletin board encourages students to refelect on their work and consolidate their learning.

```
TO TIP.TOP
 FULLSCREEN TORCH LHAND LARM CROWN EYES NOSE MOUTH HAIR NECK CHIN BOOK RHAND DRESS LOD
END
```



```
TO S :X :Y
 SETXY :X * 17 :Y * 14
END
```

```
TO TORCH
 HT
 PU S - 2.5 8.5
 PD S - 2.8 8
 S - 3 8
 S - 3.2 7.5
 S - 3 7
 S - 2.2 7
 S - 2 7.5
 S - 2.2 8
 S - 2.5 8.5
 PU S - 2.4 5.8
 PD S - 2 5.9
 S - 1.9 6
 S - 1.95 6.6
 S - 2 6.7
 S - 2.1 6.5
 S - 2.3 7
 S - 2.5 6.7
 S - 2.6 7
 S - 2.7 6.7
 S - 2.9 7
 S - 3 6.7
 S - 3.1 7
 S - 3.2 6.7
 S - 3.3 6.9
 S - 3.5 6.5
 S - 3.6 6.55
 S - 3.6 6.4
 S - 3.5 6
 S - 3.4 5.8
 S - 3.1 5.6
 S - 2.4 5.8
 S - 2.4 5.3
 PU
 S - 2.5 4.7 PD S - 2.5 3.7
 S - 3 3.6
 S - 3 4.5
 PU
 S - 3 5.2
 PD S - 3.05 5.7
END
```

Figure 7.8. Heather and Joanie's Final Result.

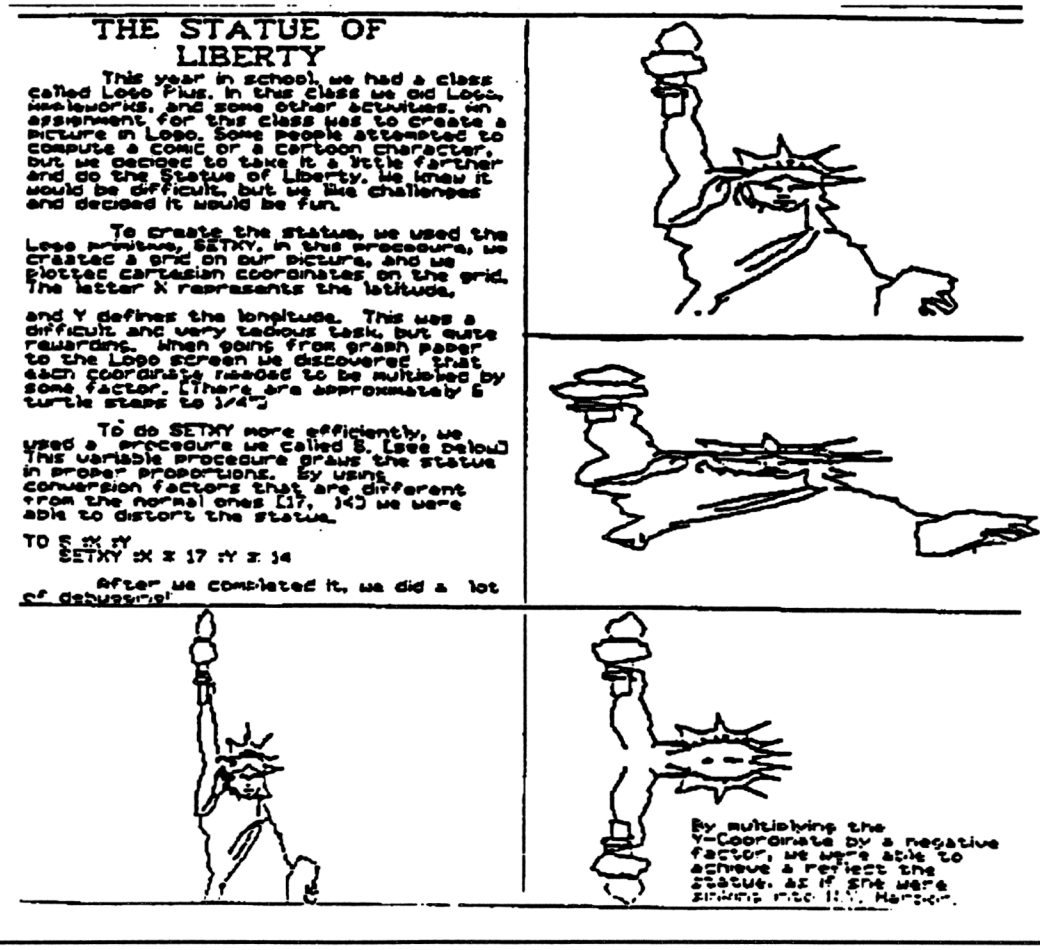


Figure 7.9. Heather and Joanie's Published Results

Some Concluding Remarks

We focused this entire chapter on just one project because Heather and Joanie's opus illustrates almost all of the phases of the Logo programming process we have described and because their complete start-to-finish data were available. The highly structured approach their teacher encouraged is certainly not the only way to support project development, and that approach might be more appropriate for some students and some projects than for others. Nevertheless, we believe that by providing a framework that encouraged students to plan their work carefully, to keep records of all stages of their work, and to publish their results, Heather and Joanie's teacher enabled them to get as much educational value as possible from their Logo experience.

Complete as it is, however, our story of Heather and Joanie's project development is missing an essential element. Projects don't typically come to life full blown. Our data does not tell us anything about how the girls came to choose their project or about the exploration, brainstorming, and "messing around" that preceded the work described here. Such preliminary thinking, comparable perhaps to the prewriting phase of the writing process, is just as important as the phases we have described here, and it should not be ignored in attempting to support students in project development.

Finally, we feel the need to say something about the particular approach Heather and Joanie used to draw their statue. At first we were somewhat uncomfortable that conventional turtle graphics and readable Logo procedures were abandoned for the much less intuitive process of plotting points and connecting them. However, when we realized that the girls had been using standard turtle graphics effectively for almost three years, when we saw all the new types of mathematical thinking that this approach involved, and when we took into account the excitement that Heather and Joanie felt at using their new knowledge to create a result that they deemed superior to what they could have accomplished using FORWARD, BACK, RIGHT, and LEFT, we were convinced of the usefulness of Heather and Joanie's approach. Even as ardent turtle-o- philes, we have to admit that there are times when coordinate systems can be just as powerful as turtles in producing excellent results and outstanding learning experiences.

Chapter 8
Managing Workspace and Files

Levels of Critical Aspect 8
Managing Workspace and Files

A. Saving Procedures on Disk in an Undifferentiated Workspace

B. Saving Work in More Than One File

C. Distinguishing Between File Names and Procedure Names

D. Using Logo Commands to Keep Track of Work

E. Making Journal Notes of Hard-Copy Printouts of Workspace Information

F. Cleaning Up Workspace and Files

Two Sad Stories

We did not originally consider workspace and file management to be a critical aspect of Logo learning. It was the teachers in the project who insisted that it was an important topic to include. They had observed that students who kept better track of their work made more progress in other areas of Logo. It was as simple as that. Unfortunately, we found that few teachers knew how to help their students learn to keep track of their work.

Sad Story 1. Once we asked a fourth-grade teacher whether her Logo students kept journals, files, printouts, or any other off-computer records of their work. "We don't have to do that in fourth grade," she explained. "In third grade, students don't write procedures, so they keep a journal to record what they have done. But in fourth grade they do learn to write procedures and save them on disk. This is much easier for them and takes much less time than keeping written records."

Sad Story 2. This same teacher taught in a self-contained classroom with five computers. Four of the computers were connected to a single printer through a switch box. We asked the teacher if she could show us any examples of her students' Logo work, pictures, or printouts. She apologized that she did not know how to print copies of Logo procedures or pictures. "What do you use the printer for?" we asked. "To print out their compositions when they use word processing," she explained.

There is an interesting history behind this story. Logo was the very first elementary computer application this school district had used. From the beginning, the district decided to put a computer in every classroom in Grades 3-5 but could not afford to purchase printers to go with them. An elementary Logo curriculum was written that entirely ignored the use of printers! Later, when word processing was added to support writing instruction, more computers were purchased, along with one printer per classroom. The teachers were trained to use the printers for word processing but not for Logo. So what began as a reasonable compromise—to stretch computer resources as far as possible—deteriorated into a poor teaching practice. When printers were added, teachers were not shown how to use them with Logo, nor were they expected or encouraged to do so.

What's Critical About Workspace and Files?

In his famous polemic against our society's growing dependence on computers, *Computer Power and Human Reason*, computer scientist Joseph Weizenbaum (1979) argued that when computers made it possible to store and retrieve more data than could ever be kept in a set of physical file drawers, social organizations had grown too large for human beings to manage effectively and thereby had become both impersonal and inefficient. Logo was designed to do the opposite, that is, to make computers more amenable to human control and understanding. And yet, when information is stored invisibly on magnetized disks and when graphic images disappear—perhaps forever—when the power is turned off, the resulting confusion about what's been created and how it can be re-created can be as frustrating and dehumanizing as any problem with a computerized airline reservation or telephone bill.

Imagine a file drawer without any folders—just a huge pile of papers of different sizes, scraps, notes, bills, and so

forth, all stuffed in with no organization whatsoever. Or imagine a loose-leaf notebook without dividers. Suppose that every time you write something, you add it at the end of the notebook, but whenever you use the notebook again, all the pages have been rearranged so that you can't remember what's there or the order in which the entries were written. That's what a Logo workspace or disk can be like for someone who doesn't know how to organize it.

Unlike using 360 degrees in turtle geometry or using subprocedures in programming, understanding file and workspace management is not important because the intellectual ideas are especially powerful or especially accessible through Logo. Rather, the ideas and strategies of workspace management have a *negative* power if you don't understand them. Put another way, if you do not manage your workspace and files effectively, you are likely to waste tremendous amounts of time searching for and re-creating past work, and you will miss out on many significant learning opportunities simply by not having easy access to the work and the thinking you have already done.

As far as we can tell, there's only one powerful idea associated with file and workspace management: If you're working with ideas and information, it's a good idea to organize them so that you know what you've got, when you created it, and how you can find it again. Some of this can be done using the computer itself; however, we find that hard-copy printouts and a notebook or journal are essential for most people to keep track of their work. In the sections that follow, we describe several stages that we have observed students of all ages go through as they learn to keep their work organized.

A. Saving Procedures on Disk in an Undifferentiated Workspace

This is an almost universal starting point—whether a student is using *LogoWriter*, which utilizes a page metaphor, or a version of Logo that uses SAVE and LOAD or READ to save and recall files. The natural tendency is to ignore the complexities of file management at first and save a bunch of unrelated exploratory procedures on the same page or workspace. An advantage of this is that a Logo beginner is free to concentrate on learning to use the turtle and to write procedures and subprocedures without having to worry about naming files and recalling them correctly.

Subtle differences between versions of Logo may influence how soon a learner breaks away from this extremely simplistic approach to saving work on disk. For instance, Terrapin Logo makes it easy to keep saving work in the same file over and over again. So does *LogoWriter*, with its automatic save feature that saves files whenever you leave the page you're working on. Other LCSI versions of Logo won't let you save new material in an old file unless you erase the previous version of the file. This probably leads students to use multiple files at an earlier stage of their work.

B. Saving Work in More Than One File

Once a student begins work on one or more specific projects, it is important to save work in more than one file. At first, a student might simply give each new file a slightly different name. Brad, a fourth grader, worked for many weeks on a city project and changed his file names several times as he worked. His names included CITY; CITY.2; GOOD.CITY; BIG.CITY; BIG,CITY (possibly a typo); and so on. All of his file names were also names of procedures in his workspace. Brad kept no written records about what was in these files and soon became confused about which one was the most up-to-date. Sometimes he would load in several different files when looking for a particular version of his project, and then save the entire workspace using just one of the file names.

Two strategies might have helped Brad. First, he could have been encouraged to save files with sequential names: CITY1, CITY2, CITY3, and so on. In this way, he would know that his earlier work was still available and that the file with the highest number was the last work he had done. It would have been even better if he had kept a journal and had written down the names of new procedures, the dates they were written, and the file they were saved in.

C. Distinguishing Between File Names and Procedure Names

Realizing that *all* your current procedures are saved when you save a file or a page is a key understanding for effectively managing files and workspace. Brad may have been suffering from a very common misconception among beginning Logo users. He may have thought that he needed to save each procedure he wrote in a file with its own name and may not have been aware that all his procedures were saved every time he used the SAVE command. Ironically, some teachers (and even one early Logo textbook) reinforce this misconception when they teach students to save each new procedure in a file with its own name as a way of making sure that a student's work is saved continuously as he or she works.

LogoWriter users may be less likely to hold this misconception. It is clear that an entire *LogoWriter* page is saved every time you leave the page. And whenever you move to the Flip Side of a page, you are aware of all the procedures written on it. This feature makes other aspects of workspace management more difficult, but it is extremely useful in helping students avoid the procedure name/filename confusion.

D. Using Logo Commands to Keep Track of Work

Most versions of Logo (other than *LogoWriter*) include special primitives that help users understand the contents of their workspace and disk. For example, students can be encouraged to type POTS (printout titles) before saving a file. This would help them see that all the procedures listed were being saved every time. If they want to see the text of a specific procedure, they can type PO CITY or PO "CITY (depending on the version of Logo they are using).

LogoWriter does not have these features. A user who wants to know exactly what procedures are on a particular page must go to the Flip Side and page through several screens to see all the procedures written there. However, *LogoWriter* users can take advantage of cut-and-paste functions to reorganize their procedures in a convenient or logical order and can easily add lines of text explaining what particular procedures do or when they were written. Another advantage of the way *LogoWriter* does business is that procedures stay where you put them, so that if you can remember where you placed a procedure on the Flip Side, you can find it again rather easily.

The CATALOG command shows all the files on a particular disk. The *LogoWriter* Contents Page performs the same function much more naturally, so no special command is needed.

E. Making Journal Notes or Hard-Copy Printouts of Workspace Information

If students have a printer available during their working sessions, they can print out the current contents of their workspaces, procedures, and disks and save them in a file folder or paste them into a notebook or journal. Although time-consuming, this may be the best way to keep track of work done on a day-to-day basis. It's fairly simple to direct output to a printer and then use POTS, PO (followed by a list of procedures), PO ALL or POALL, or CATALOG to print the information needed. PO ALL or POALL prints out all procedures and variable names in a workspace and may generate more paper and more data than is useful. It also prints procedures in a random order. It is often more useful to use PO followed by a list of procedures to print out only those procedures that have been changed in a given session or to print a particular set of related procedures in some useful order.

LogoWriter users can print the entire Flip Side of a page by typing PRINTTEXT or PRINTTEXT80 from the command center, but it is more difficult to print out individual procedures. Also, *LogoWriter* has no POTS command, and there is no way to print out the Contents Page unless your printer is equipped with a Printscreen button.

If printers are not available, notes in a journal are the next best thing. Students may want to copy the list of procedures generated by POTS and the list of files generated by the CATALOG command.

F. Cleaning Up Workspace and Files

One of the last workspace-related skills most Logo users learn is cleaning up workspace and disk files. "Cleaning up" means using the ERASE command to eliminate unnecessary procedures from a workspace and saving it in a less cluttered form, and using the ERASEFILE command to erase unneeded files from a disk. Students who regularly use POTS and CATALOG to track their work are more likely to realize that a particular procedure or file is no longer needed. *LogoWriter* users can erase an unneeded procedure by selecting its text and using the Delete key, or they can delete an unneeded page from the Contents Page by selecting it and using the Delete key.

It's a bit more complicated when you want to clean up a file by dividing it into two or more separate files. In such a case you'll need to save the file twice, with different names, and erase different procedures from each file.

Why Is Workspace Management So Difficult to Teach?

We think there are three reasons why teachers find it difficult to support their students in managing their Logo files and workspaces effectively.

First, many teachers have not learned the concepts and skills of workspace management themselves. Many Logo training sessions for teachers, especially ones that consist of one or two after-school sessions designed to cover what their students are expected to learn, do not cover specialized commands such as POTS, PO, and ERASE, nor do they explain how to print hard copies of Logo procedures and graphics.

Second, many teachers are afraid that if they teach their students how to erase procedures or files, that essential work may be lost. This is a very real concern! Most computer users have had similar experiences at one time or another, and we want to protect our students from having the same frustrations we have had. One solution is for teachers to back up files regularly. (As teachers, we often do this for our students, making back-up copies of student disks once a week or so and storing them in a safe place.)

Finally—and we think this is the biggest problem—because computers are a scarce resource, we all want to maximize the benefits of computer access for our students. We want our students to devote as much of their time as possible to creative work and problem solving, so we don't require that they keep notes or make printouts because of the time and mechanical effort involved. What we often don't realize is that a huge amount of time is wasted when students try to find old work, remember what a particular procedure does, or what file last week's work was saved in. A few minutes routinely spent in keeping records of each session can save hours in the long run. In addition, students who make printouts of their work can examine and think about it while they are away from the computer, and this might make their computer time more valuable and effective.

Assessment Is Important for Students as Well as Teachers

We've said throughout this book that ongoing assessment of student learning is an essential ingredient of effective Logo teaching. In order to assess Logo work, teachers need data—especially printed examples of the procedures and graphics that their students are creating. If students learn to keep track of their own work and to print out procedures regularly, it could make their teacher's assessment work much easier.

But students also may directly benefit from keeping track of their work, especially if they keep printouts of their

procedures and pictures. Students who examine their previous work as they think about what to do next may very well become more effective problem solvers. Students who habitually keep track of their work have the opportunity to assess their own Logo progress, to become aware of how their own thinking is developing. Helping learners to think about thinking was one of the fundamental goals of Logo's developers. To the extent that it contributes to that objective, workspace and file management may involve powerful learning after all!

Chapter 9
A Reflective Teacher's Interventions With a Tentative Learner of Logo— A Case Study

More than 20 years ago, a science educator and philosopher named David Hawkins (1970) gave a talk to some primary teachers in Leicestershire, England. His talk, entitled "I, Thou, It," has become a classic reference point for educators. Hawkins explored the relationship between a teacher, a child, and "a third thing which has to be there" (p. 45). He made the point that children can learn complicated things "where there are human beings who serve, so to speak, as part of the learning process." He goes on to specify that "the function of the teacher, then, is to respond diagnostically and helpfully to a child's behavior, to make what he considers to be an appropriate response, a response the child needs to complete the process he's engaged in at a given moment.... The teaching isn't going to go on forever: it's going to terminate at some point in the future. ...The child should learn to internalize the function which the adult has been providing" (p. 47).

This chapter presents the first of two case studies showing how teachers participating in the Logo Action Research Collaborative (LARC) have used assessment of student work to inform their teaching. These studies show examples of Logo teachers' attempts to answer their own questions about how to provide structure and support for particular students in creating their chosen Logo projects. Our collaborative researchers are experienced classroom teachers with at least two years of prior Logo teaching where students are working in a Logo environment. Part of our work supports the teachers' professional development by helping them identify their own research questions and design and carry out their own classroom action research investigations.

We ask the teachers to choose one of their students that they are particularly interested in for their own reasons, and to choose a second student who offers a contrast in some way to the first student. Teachers record day-to-day observations of the students, formulate their own research questions, and prepare a report on their observations and the effects of their interventions.

A Case Study of Diagnostic Teaching Practice

This case study documents David Crump's development as a diagnostic Logo teacher as we follow his observation of a fourth-grade student we call Kathy. The study is based on data David collected from the student, his written observations about her work and his interventions, a research report he wrote, formal interviews conducted with David by the project staff at the beginning and end of the research project, and information shared at project research meetings. You will recognize Kathy's work from examples we used in previous chapters to illustrate students' use of procedures and subprocedures and of mathematical thinking. David eventually published his own description of this research in a Boston-area newsletter for computer-using educators (Crump, 1990).

When David was interviewed at the start of the project, he was asked, "What is your belief about Logo's importance and benefit to your students?" David responded, "I have a big question mark in my mind about that." When asked "Have you identified any specific problems at specific stages of Logo learning?" he replied, " I'd have to say that when children find out about writing superprocedures and calling procedures, I see an extra amount of excitement in their being able to use Logo. When they find that they can actually apply that, it sticks. I think that's a big breakthrough." When asked "What is your belief about how children learn Logo best?" David replied, "My belief is that the more experience they have using Logo, the better they learn to use Logo."

David chose Kathy as one of his two students because he was concerned that mid-way though her second year of Logo experiences she was not making adequate progress nor was she using the ideas and strategies he was teaching her. David reported that Kathy was "considered to be rather average in overall ability. I considered her ability to work in the Logo environment to be one of the lowest in the class. She was always very tentative in her approach to using Logo, and she always solicited a great deal of help from me as she worked on her Logo projects. It was my hope that by focusing attention specifically on her, I would be able to help her overcome some of her problems with Logo."

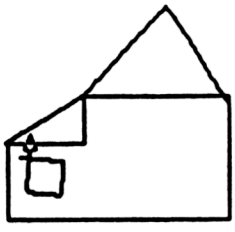
David's own research questions were closely related to what he was trying to learn by following Kathy's progress, and this informed his observations throughout the year. His questions were:

- How do students use what they are given in teacher-directed lessons?
- Is there a change in programming style over time?
- How much help do students solicit and what type of support do I have to provide?

In David's school system, the Logo curriculum starts in third grade: students practice moving like a turtle and use

basic turtle commands to move the turtle around the Logo screen. In fourth grade, children are taught to write procedures and subprocedures as part of a curriculum focus called "Explore or Create."

In Figure 9.1, Kathy's HOUSE.6, is a procedure named HOUSE.6 that Kathy wrote in early March of her fourth-grade year, fully a year and a half after she started to use Logo.



```
TO HOUSE.6
PU
BK 30
BK 20
RT 90
FD 30
FD 20
LT 90
PD
FD 50
FD 10
FD 20
LT 90
FD 65
FD 30
LT 90
FD 30
RT 90
FD 30
FD 20
LT 90
FD 30
FD 10
FD 5
FD 4
LT 90
FD 56
FD 56
FD 20
FD 10
FD 3
PU
LT 90
FD 56
FD 30
BK 5
BK 4
BK 3
LT 45
RT 5
RT 3
RT 4
PD
FD 20
FD 30
FD 20
FD 5
LT 103
FD 80
RT 15
FD 60
PU
FD 4
LT 90
LT 29
LT 10
FD 20
LT 10
LT 5
BK 5
PD
FD 10
FD 8
FD 5
FD 2
FD 4
RT 90
FD 6
FD 4
FD 5
FD 7
RT 90
FD 10
FD 7
FD 3
FD 3
RT 90
FD 3
FD 2
FD 2
FD 5
FD 5
FD 6
FD 3
FD 1
FD 1
END
```

Figure 9.1. Kathy's HOUSE.6

When David brought a printout of HOUSE.6 to share with our research group, he seemed to be somewhat discouraged about Kathy's work. He reported that he had been teaching superprocedures and subprocedures since early in the year, and Kathy's work did not seem to reflect that part of his teaching at all. The entire research group worked together to understand as much as we could about Kathy's work. Starting with a printout of Kathy's picture and procedure, our group brainstormed answers to three questions:

1. What knowledge is Kathy using?
2. What potentially useful knowledge is Kathy not using?
3. What teaching interventions would you, as Kathy's teacher, take to support her in using Logo more powerfully?

We spent more than an hour examining the printout shown in Figure 9.1. We make a point of not overlooking obvious or seemingly trivial knowledge so that we can get as complete a picture as possible of the student's understanding. We discovered that Kathy was using a lot of Logo knowledge in her work (more than David had realized in his own first assessment). The knowledge Kathy was using at this point of her work includes:

- something about what a house looks like
- basic turtle commands
- how to use a turtle to create a picture
- how to use PENUP and PENDOWN to place the turtle without tracks
- how to position the turtle on the screen to center a drawing
- how to use 90 degrees to make square corners
- how to estimate angles that are not 90 degrees
- how to observe the turtle and make fine adjustments
- how to solve problems by trial and error, making successive approximations
- how to to use BACK and FORWARD as inverses
- how to use RIGHT and LEFT as inverses
- how to write and edit a simple procedure.

We next identified some knowledge Kathy hadn't used but might find useful, an important intermediate step to take before deciding on a teaching intervention. Kathy might find it useful to know:

- how to position the turtle so that the window in her house would look square
- how to combine steps (to make her procedures easier to read and to identify patterns)
- how to use REPEAT (for a rectangle or square)
- how to use subprocedures to divide her work into smaller pieces.

It could be be daunting to any one of us to receive feedback on all the aspects of our work that we might change. Therefore, as teachers, we focus on a short list of possibilities in order to find one or two that might make a real difference to the learner in solving her own problem and that fits the problem-solving approach she is already using.

Our next challenge is to find acceptable, nonthreatening ways to share our knowledge with a student so that he or she may choose to use it. At this point we almost always go to a student to check out our observations and interpretations before initiating any intervention at all. Let's see what David did. Here are excerpts from his written observations:

> 3/13 OBSERVATION: The basic approach of both Kathy and her partner has been to work in the immediate mode, one partner writing down all of the steps

that seem to work, the other typing in the commands. After they get so far, they switch to the editor and enter these same commands into one long procedure. What is interesting is that absolutely no attempts have been made to compensate or make adjustments for any slight errors made while investigating in the immediate mode. An example of this is not changing FD 10 FD 20 FD 5 FD 2 FD 3—typed in the immediate mode—to FD 50 in the editor.

Kathy did solicit help during the observation period.... Kathy was not satisfied with the way her window came out, and she couldn't figure out how to make it look like a rectangle. She said that it also kept coming out looking a little lopsided. I asked her what she wanted to do and she said she wanted to make a normal-looking rectangle for a window. She didn't seem to be aware that the turtle was turned to perhaps 2 degrees right before she made the window. I suggested that she check the turtle's heading [by using the command PRINT HEADING]. She saw then that the turtle's heading was off a bit and then actually asked me what she should do to fix it. I told her that she might try making the turtle's heading 0 before making the window. It took her two tries before she got it right.

David's first comment focused on his observation that Kathy and her partner did not combine steps or eliminate unnecessary ones when moving from the immediate mode to the editor. His second observation shows how he talked with Kathy and found out that she was dissatisfied with her window: she wanted it to look like a rectangle and not be "lopsided." When Kathy asked for help, he was ready to show her one new piece of Logo knowledge and how it would help her solve her problem. Kathy could not have discovered this information; someone needed to make the intervention. Once Kathy knew that there was a way to find the turtle's precise heading and then make it exactly 0, she could solve her problem.

David's observations continue:

3/16 OBSERVATION AND INTERVENTION: I felt compelled while watching Kathy today to say something about writing her commands in a more concise way when taking commands from immediate mode and putting them in the editor. She almost immediately complied but did not wish to go back through the entire procedure and make the necessary changes. I told her it was necessary from this point on that both she and her partner make a more serious effort to do this in every procedure they made. Kathy was just beginning to type in the commands that made the rectangular window when I heard her say to her partner, "Let's use a REPEAT to make this." They got the turtle in position to make the door and helped each other with a REPEAT that made the door. They started talking about making two square windows and where they should place them when the time was up.

3/18 and 3/20: Kathy was out sick.

3/23 OBSERVATION: Kathy and her partner were able to make the two square windows they had talked about making. As I looked at their procedures, I noticed that when they made the square windows, they were not written as REPEATs. They tried placing the windows in three different locations before they seemed satisfied with what I hope is their final location.

On 3/16, David observed that Kathy knew about the REPEAT command. However, after missing school for a week, she did not use REPEAT even though it would have been appropriate for what she was doing and might have made her work easier. She did, however, continue to combine small steps before entering them into the editor and eventually completed HOUSE.6, shown in Figure 9.2.

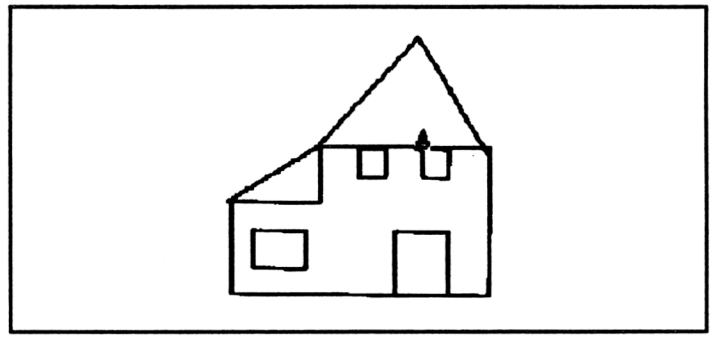


Figure 9.2. Kathy's Completed HOUSE.6 (Grade 4)

Later, David recorded these remarks:

On 3/27 I conducted a thorough review of the method of planning, sketching, and then deciding on subprocedures that will be used.... I then explained the next project that I wanted them to work on. It involved planning with their partner, sketching, and using super- and subprocedures to make a picture of their own choosing.

David went on to record Kathy and her partner's decision to draw a schoolhouse, a sketch of which is shown in Figure 9.3, Kathy's Schoolhouse.

David recorded his reminders to them at several points to combine steps such as RT 30 and RT 60 into RT 90. Before Kathy and her partner went to the computer, David reviewed the idea of using a superprocedure, using the same examples he'd used in the whole-class lesson. These teacher interventions were different from the earlier ones in that they were made *before* rather than after the students started programming.

David's work with Kathy made him pay closer attention to the Logo knowledge his other students were and were not

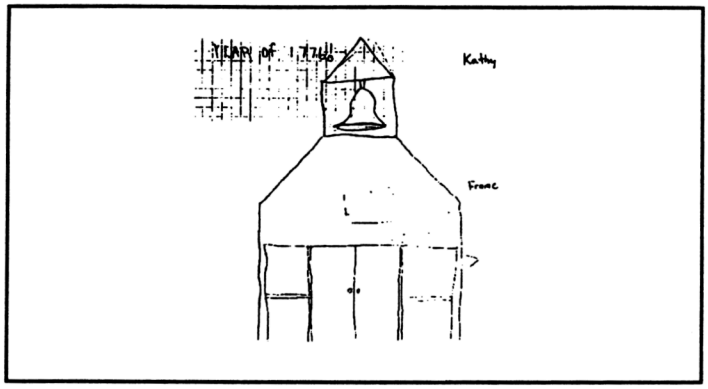


Figure 9.3. Kathy's Schoolhouse

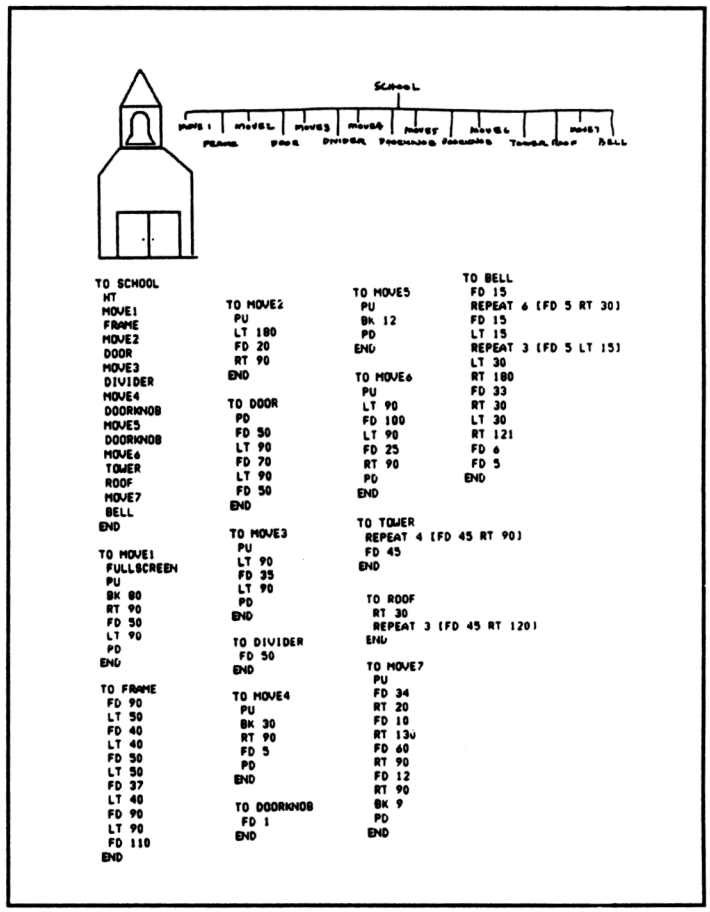
using. On 4/8 he recorded a whole-class "Intervention Prior to Next Assignment," from which we have taken the following excerpt:

> I noticed on 3/25 that only about half the students in the class were attempting to use super- and subprocedures when working on Logo projects of their own choosing. When I asked the class about their reasons for continuing to write the long "spaghetti" way, I was basically told they felt more comfortable doing it that way. There were about six students who said that even though I had explained it before and even made them try using them, they still didn't understand it. I explained to them that I felt they were ready to start using this "higher level" approach to Logo programming and problem solving, and we decided as a class some review might be helpful. I honestly believe that if most, or even all, of the students are going to begin using this method of writing procedures, intervention now is necessary.

As Kathy worked on her schoolhouse project over the next month, she incorporated many of David's suggestions. Working from a sketch, she labeled and named each part of the drawing and wrote separate subprocedures for each part. She also used REPEAT to draw regular geometric shapes. David's attention shifted to other aspects of her work. Here are his observations of Kathy's work recorded on 5/4:

> I honestly don't believe she needs as much help as she asks for. I asked her why she always asked for help. She said she wanted to make sure she was doing things the right way and didn't want to make too many mistakes.... I told her I didn't mind trying to help them, but I wanted them to try something several times before they decide they are stuck and really need some assistance. I said they could ask me for help if they wanted to make something and really didn't even know how to even start making it.

David's notes tell us that after several more interventions from him, Kathy and her partner went on to complete their schoolhouse with a rather sophisticated bell shape. His interventions were more subtle than previous ones. They were given in the form of questions and suggestions: "Can you find a way to draw a smaller circle by changing the numbers in your procedure and watching what happens?" "Can you figure out how to make just half of a circle? Then you will know almost all you need to make the bell!" Kathy's final result is shown in Figure 9.4.



```
TO SCHOOL
 HT
 MOVE1
 FRAME
 MOVE2
 DOOR
 MOVE3
 DIVIDER
 MOVE4
 DOORKNOB
 MOVE5
 DOORKNOB
 MOVE6
 TOWER
 ROOF
 MOVE7
 BELL
END

TO MOVE1
 FULLSCREEN
 PU
 BK 80
 RT 90
 FD 50
 LT 90
 PD
END

TO FRAME
 FD 90
 LT 50
 FD 40
 LT 40
 FD 50
 LT 50
 FD 37
 LT 40
 FD 90
 LT 90
 FD 110
END

TO MOVE2
 PU
 LT 180
 FD 20
 RT 90
END

TO DOOR
 PD
 FD 50
 LT 90
 FD 70
 LT 90
 FD 50
END

TO MOVE3
 PU
 LT 90
 FD 35
 LT 90
 PD
END

TO DIVIDER
 FD 50
END

TO MOVE4
 PU
 BK 30
 RT 90
 FD 5
 PD
END

TO DOORKNOB
 FD 1
END

TO MOVE5
 PU
 BK 12
 PD
END

TO MOVE6
 PU
 LT 90
 FD 100
 LT 90
 FD 25
 RT 90
 PD
END

TO TOWER
 REPEAT 4 [FD 45 RT 90]
 FD 45
END

TO ROOF
 RT 30
 REPEAT 3 [FD 45 RT 120]
END

TO MOVE7
 PU
 FD 34
 RT 20
 FD 10
 RT 130
 FD 60
 RT 90
 FD 12
 RT 90
 BK 9
 PD
END

TO BELL
 FD 15
 REPEAT 6 [FD 5 RT 30]
 FD 15
 LT 15
 REPEAT 3 [FD 5 LT 15]
 LT 30
 RT 180
 FD 33
 RT 30
 LT 30
 RT 121
 FD 6
 FD 5
END
```

Figure 9.4. Kathy's Completed Schoolhouse (Grade 4)

In his final report, David stated that after the conversation about trying things herself, Kathy was a changed person. She did continue asking for his help, but with less frequency and only when she had tried and still couldn't figure out what to do. "I saw her making mistakes and working on them until she had resolved them. I felt I had finally made the big breakthrough with her that I was hoping for," he reported.

The final project for the class was a procedure to draw a toothbrush and then spin the brush. David recalls Kathy's new-found confidence:

> During the final project of the year, Kathy didn't ask for any help at all. There were times when I wanted to help her with some of the mistakes she was making, but she was working most of them out on her own. The school year finished before Kathy could complete the Toothbrush and SPIN.BRUSH project, but I was pleased to see she finally felt confident enough in the entire project without any help from me.

Kathy's SPIN.BRUSH Project is shown in Figure 9.5.

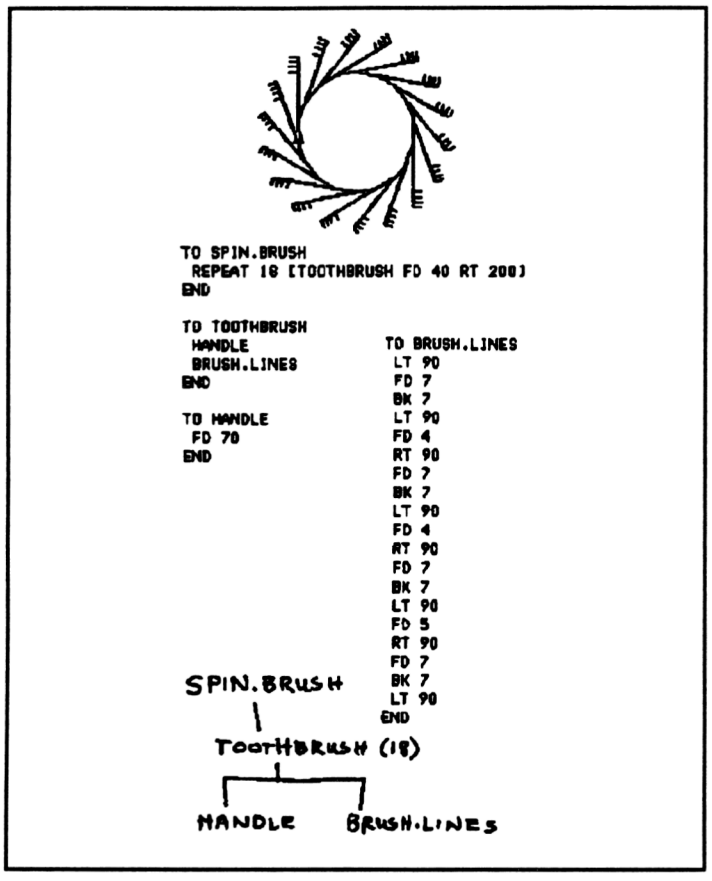


Figure 9.5. Kathy's SPIN.BRUSH Project

Conclusion

David Hawkins (1970) believed in educating children in a rich, diverse, active environment, with many opportunities for teachers to observe students at work. These observations would allow a teacher to collect information about students and become diagnostic teachers. In this way, he suggested that you can "refine your behavior as a teacher far beyond the point of what's possible when every child is being made to perform in a rather uniform pattern."

As we have seen, David Crump succeeded in creating an "I, Thou, It" relationship with Kathy, whose "ability to work in the Logo environment," he considered to be "one of the lowest in class." He accomplished this change in his relationship through being a diagnostic teacher and investigating his own questions in his own classroom.

During the project, David developed a more differentiated, expanded view of how students learn Logo best. When asked about this at the beginning, he said simply, "The more experience they have using Logo, the better they learn to use Logo." At the end of the project, when asked about how children learn Logo best, he replied,

> First, by being given good instructional models. Second, by having the opportunity to experiment, to try to use what they have been shown.... Third, I think it's very important for kids to have the time to do whatever they want to do, regardless of what they've been taught, so that they become part of that Logoland out there.

When asked at the end of the project about targets of difficulty for his students, David traced the process by which he had encouraged his students to use superprocedures and subprocedures. The turning point came when he led a discussion with his students about the benefits of this approach. He recounts:

> [My problem] centered around getting kids to see the benefit of breaking down their procedure into its component parts, superprocedures and subprocedures. After the first lesson, some kids wouldn't buy it. They resisted, would not do it. They were given a lesson where they had to do it, and then there were still some kids who didn't want to do it, but did it anyway. I would say, "In the next project, I would like you to use subprocedures and superprocedures." Half the kids used it. Half the kids didn't. That's when we had a class discussion about what some of its benefits are. The most important thing for them was that when we share, you can ask people to show a part of their program, you can see it, rather than saying, "Well, I can't show you that because I don't know where it is." And that was an important reason for them.

Through professional engagement in classroom action research, David transformed an experience that could have been merely confusing or frustrating into an opportunity to investigate his own questions and to make sense out of his own teaching and the learning of students in his classroom. In this example, we have seen how David engaged Kathy in working more effectively and powerfully with Logo. By sharing his documentation and insights with us, he made a professional contribution to the wider community of Logo-using educators.

Finally, this experience allowed David to come to grips with his own concerns about the place of Logo in his fourth-grade curriculum. Although he continued to have doubts about some of the specific content and time allotted in his school district's Logo curriculum, he concluded his report about his personal research project by saying:

> I am now able to justify in my own mind why the teaching and learning of Logo is important. If taught the right way, students learn to think both logically and creatively. Not only do they have the opportunity to learn about math, problem solving, and programming, but more importantly, they begin to learn about their own thinking.

Chapter 10
When Teacher *and* Student Are Stumped

Our national agenda for school reform places an increasing emphasis on the teaching and learning of problem-solving skills and on the professional development of teachers. Yet most teachers have had little, if any, education or training in teaching problem solving. In most of our teaching we already know the solutions to problems we give our students. Then we teach them the specific steps and procedures involved in solving those problems. Yet, real-world problems do not have known solutions—they may even be unsolvable. And real-world problem solvers use methods that may lead to dead-ends, forcing them to start over. Unfortunately, as teachers, we have not had many opportunities to solve mathematical problems that are difficult to us. Our training does not give us much experience in teaching students to solve problems whose solutions are not already known.

In *Educating the Reflective Practitioner*, Donald Schon (1987) states: "The practitioner must choose. Shall he remain on the high ground where he can solve relatively unimportant problems according to prevailing standards of rigor, or shall he descend to the swamp of important problems and non-rigorous inquiry?" (p. 3). Schon goes on to say, "But, as we have come to see with increasing clarity over the last twenty or so years, the problems of real-world practice do not present themselves to practitioners as well-formed structures. Indeed, they tend not to present themselves as problems at all but as messy, intermediate situations" (p. 4). Schon eventually asks, "Can the prevailing concepts of professional education ever yield a curriculum adequate to the complex, unstable, uncertain, and conflictual worlds of practice?" (p. 12).

A programming environment that allows students to work on problems of their own devising can simulate problem solving in the real world. A project that students have created becomes "real world" for them. The student or the teacher often does not know solutions or pieces of the solution. Frequently, the problems can become so complicated and intertwined that neither the teacher nor the student knows where to start. The situation may be even more uncomfortable for teachers when their students surpass their own comfort level in programming. How can teachers model problem-solving strategies when they themselves don't know the answer and thus cannot intervene effectively to help students solve problems?

Perhaps by thoroughly analyzing the work of a few students, teachers can get a feel for how to intervene to help students with their problem solving in Logo. As we indicated in Chapter 9, part of our work with teachers in the Logo Action Research Collaborative involves asking them to choose two students and follow their work throughout the year. We have also developed structures to help teachers with specific questions and difficulties they face, either in their research or in their Logo teaching. The following case study illustrates how the process we developed works when neither the teacher nor the student has the answer to a problem the student is trying to solve.

A Case Study of Being Stumped

Katherine (Kitsy) Rothermel, a participant in our exploratory research project, teaches Logo in a fourth-grade class in suburban Boston. Her students had Logo as third graders as well. Six computers are located in the back of the classroom, where students cluster to work on their own projects while Kitsy conducts small-group work with other students. She has taken several Logo inservice training workshops each year for six years but does not consider herself a programmer in Logo. She is extremely skilled at setting up a rich, experiential learning classroom for her students.

Kitsy chose a Logo problem of one of her students, Douglas, to share with our group of teacher-researchers examining Logo learning. She brought the problem to us because she did not know how to help Douglas solve his own problem, and she wanted the group to offer suggestions.

Douglas' problem is shown in Figure 10.1, Douglas' PENTACIRCLE.

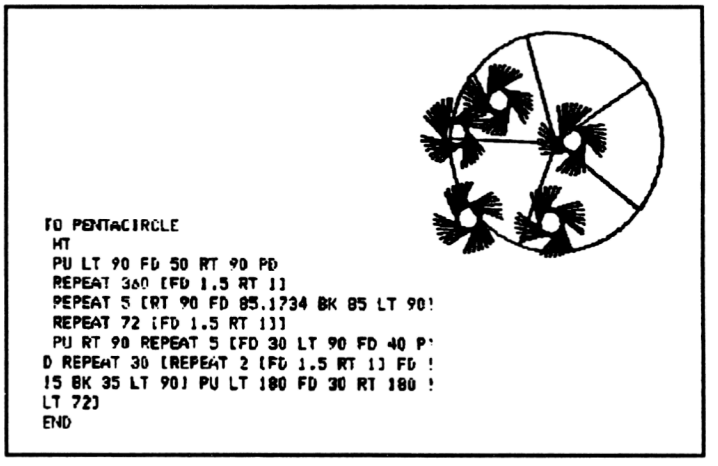


Figure 10.1. Douglas' PENTACIRCLE (Grade 4)

Douglas wrote a program that does something interesting, yet he was not satisfied with it. His procedure does not distribute the designs—which we came to call "whirligigs"—evenly in each section of the divided circle. He responded to this problem by writing his teacher a note saying, "I wish that the design would go in each part of the circle."

Our research group spent two hours attempting to help

Kitsy help Douglas. We divided into three small working-groups and set up a three-stage process.

Stage 1. Starting with a printout of Douglas' picture and procedure, each group answered three questions:

1. What knowledge is Douglas using?
2. What potentially useful knowledge is Douglas not using?
3. What would you, as a teacher, suggest to Douglas to help him solve his problem?

Stage 2. After each group brainstormed to generate lists of responses to each question, the group worked collaboratively on the computer to solve Douglas' problem. As we worked, we kept in mind his knowledge and his particular ways of working so that our solutions might help us actually coach Douglas in solving the problem. Each group documented this process.

Stage 3. The third aspect of the process was recursive: we asked ourselves the three original questions again, but this time the answers were based on our experience of attempting to solve the programming problem ourselves.

If you are feeling adventurous, we suggest you gather a group of colleagues and work through these three stages yourself before reading on.

Attacking Douglas' Problem

Here are some of the items we listed in Stage 1, answering the three questions before working on the problem:

What knowledge is Douglas using?

- REPEAT commands
- How to make a circle
- Assume he understands a decimal
- How to write a procedure
- Nested REPEATs
- Something about 85.1734
- How to move a turtle without a trail
- Geometry and division of a circle
- Knowledge of basic commands

What potentially useful knowledge is Douglas not using?

- Not using sub- and superprocedures
- Needs a MOVEprocedure before plugging in the design
- Doesn't know SETXY
- Placement of one command per line
- How to place things on the screen
- Understanding the total turtle trip
- How to debug using the TRACE command

What would you suggest to Douglas to help him solve his problem?

- Sub- and superprocedures
- MOVE procedures
- If he knows decimals, he should be able to figure out midpoints
- May go into variables to get the circle bigger so it will fit
- Redesign the entire program—make one pie wedge and design and turn it to reproduce
- Suggest use of TRACE and perhaps introduce SETXY command and SETH command

We spent the next hour divided into three groups, attempting to rework the program ourselves. One member (Jane, Bob, Kitsy) from each group served as a recorder. The programming was tough and no group finished the process, but each was engaged fully in the challenge. Each of the groups approached the problem in a different way, yet each attempted to solve it in a way that was consistent with Douglas' own approach.

Jane's group reported: "We really tried not to change too much of his program but to break it down into pieces....We had problems because we were thinking about it opposite of the way he was thinking about it. He was thinking about moving the turtle from the outside in and we were thinking about moving it from the inside out."

Bob's group reported: "Our overall strategy was to leave it basically the way it was. We introduced subprocedures and a superprocedure. We liked the way he drew the circle. We got rid of the nested REPEATs and knocked off some of the decimals. We decided he didn't really understand decimals.... We centered the circle. Our ultimate strategy to put one whirligig in each piece was to make a MOVE procedure to center the turtle in a good position."

Kitsy's group thought about how to break the program down into super- and subprocedures and then how to place a simpler, state-transparent design in each section. They also thought about giving Douglas some tool procedures to help him with his moves.

By the time we finished Stage 3 of our process, in which each group again answered the original three questions after having worked with Douglas' program, our answers were more focused on powerful ideas in Logo. For example, we came up with the following ideas.

What potentially useful knowledge is Douglas not using?

- Transparent state of turtle
- Total turtle trips as applied to complete design
- Moves as distinct from turtle drawing
- Subprocedures
- Writing understandable code

The group quickly came up with many suggested teaching strategies for Kitsy to use. Most of the ideas had to do with good programming practices—keep it simple, use smaller procedures, make the moves distinct from the drawing, use state-transparent subdesigns, and use a simpler design, such as a square, before using the whirligig shape.

Where Do We Go From Here?

Kitsy took the ideas back to her classroom to offer to Douglas. To our amusement, she reported the next week that Douglas no longer wanted to change his program—he had learned to love it the way it was. (He chose to publish this design in a class Logo book at the end of the year; our research group also chose his design to put on a tee-shirt at the end of the year.)

Does this mean that the process was a waste of time? Quite the contrary. Several teachers said that they had learned a lot about Logo programming, diagnostic teaching, and coaching students on their own projects. Kitsy reported realizing that she already knew enough strategies to help Douglas; she just hadn't realized that she did. And we all realized that examining Douglas' work as carefully as we did helped us focus on how the powerful ideas in Logo allow students to carry out projects more effectively. Also, because the same ideas make procedures more readable, their use makes it easier for students to collaborate with each other and for teachers to coach students through their difficulties. Kitsy said that she will emphasize state-transparency, superprocedures, and subprocedures with all of her students. She also had many suggestions for next steps in working with Douglas on future projects.

In this case study, we solved problems on two different levels. First, there was the mathematical problem Douglas was trying to solve. Then there was the pedagogical problem we were trying to solve as teachers: How could we help Douglas solve his problem while teaching him something more about problem solving in a situation in which we did not know the answer to the original problem ourselves? While working our way through this difficulty, we were reminded of the four-step problem-solving process devised by George Polya (1957), and we made use of several of Polya's problem-solving heuristics. For example, we struggled to find what was known and what was unknown—in the mathematical sense, in the cognitive sense (by Douglas), and in the pedagogical sense (by ourselves as teachers). We thought of similar problems that we had already solved, we found general rules to apply, we divided the problem into simpler parts and named each part. Finally, as we worked, we looked back and reflected on what we had done. Having worked through the problem ourselves, we have added it to our repertoire of Logo problems and of Logo programming difficulties and thus can be more effective as Logo teachers. Investigating this one problem in great detail supported the professional development of all the teachers involved.

Japanese mathematics education, which is often held up as a model because of the spectacular success of Japanese students on international tests, offers an interesting comparison. According to Harold Stevenson (cited by Gordon, 1987), Japanese math textbooks and teachers approach problem solving quite differently than do textbooks and teachers in the United States. Stevenson says that textbooks in Japan "do not present a group of problems with answers as do American textbooks. Instead, students get a few variations of a problem they have studied. They also get very difficult examples, which no one is expected to work correctly." Furthermore, says Stevenson, "A Japanese teacher might spend one whole day on just one problem, working and reworking it from every angle, until every student understood."

We are grateful for Kitsy Rothermel's courage in sharing a problem to which she did not know the answer. We have used Douglas' problem in many workshops to the delight (and frustration) of the participants. The process we developed to deal with it can be used to support any Logo teacher to help students meet their own goals.

References

Abelson, H. (1982). *Logo for the Apple II, Apple Logo.* New York: Byte Books/McGraw-Hill.

Abelson H., & diSessa, A. (1981). *Turtle geometry.* Cambridge, MA: MIT Press.

Bamberger, J., Duckworth, E., & Lampert, M. (1981). *An experiment in teacher development. Final report.* (NIE Grant # G-78-0219). Cambridge, MA: Massachusetts Institute of Technology, Division for Study and Research in Education.

Bissex, G., & Bullock, R. (1987). *Seeing ourselves: Case study research by teachers of writing.* Portsmouth, NH: Heinemann.

Carini, P. (1979). *The art of seeing and the visibility of the person.* Grand Forks, ND: North Dakota Study Group on Evaluation, University of North Dakota.

Carnegie Forum on Education and the Economy. (1986). *A nation prepared: Teachers for the 21st century.* New York: Author.

Carver S. (1987). Learning and transfer of debugging skills: Applying task analysis to curriculum design and assessment. In R. Mayer (Ed.),*Teaching and learning computer programming: Multiple research perspectives.* Hillsdale, NJ: L. Earlbaum Associates.

Carver S., & Klahr, D. (1986). Assessing children's Logo debugging skills with a formal model. *Journal of Educational Computing Research, 2,* 487-525.

Casanova, U. (1989, January). Research and practice: We can integrate them. *NEA Today,* pp. 44-49.

Clark, C. M. (1989) *What you can learn from Applesauce: A case of qualitative inquiry in use.* (CARN Bulletin 9C). Norwich, UK: University of East Anglia, Classroom Action Research Network.

Clements, D. (1988). Research on variables, algebra and Logo. *Logo Exchange, 6*(5-8).

Clements, D. (1988-1989). Logo: Search and research. (Series of nine columns). *Logo Exchange, 7*(1-8).

Clements, D. (1990). Logo: Search and research. Strategies for solving turtle geometry problems. *Logo Exchange, 9*(4), 32-34.

Cochran-Smith, M., & Lytle, S. (1990). Research on teaching and teacher research: Issues that divide. *Educational Researcher, 19*(2), 2-11.

Crump, D. (1990). A Logo project: A tale of two case studies. *The Schoolhouse Monitor, 6*(1), 1.

Duckworth, E. (1987). *"The having of wonderful ideas" and other essays on teaching and learning.* New York: Teachers College Press.

Evans, C., Stubbs, M., Duckworth, E., & Davis, C. (1981). *Teacher-initiated research: Professional development for teachers and a method for designing research based on practice. Final Report.* (NIE Grant #NIE-G-80-0162). Cambridge, MA: Technical Education Research Centers.

Friendly, M. (1988). *Advanced Logo: A language for learning.* Hilllsdale, NJ: Lawrence Erlbaum Associates.

Gable, R., & Rogers, V. (1987). Taking the terror out of research. *Phi Delta Kappan, 68,* 690-695.

Gordon, B. (1987). Cultural comparisons of schooling. *Educational Researcher, 16*(6), 4-7.

Graves, D. (1983). *Writing: Teachers and children at work.* Portsmouth, NH: Heinemann.

Harper, D. (Ed.) (1989). *Logo theory and practice.* Belmont, CA: Brooks/Cole.

Harvey, B. (1985). *Computer science Logo style. Intermediate programming.* Cambridge, MA: MIT Press.

Harvey, B. (1985). *Computer science Logo style. Volume 2: Projects styles and techniques.* Cambridge, MA: MIT Press.

Harvey, B. (1985) *Computer science Logo style. Volume 3: Advanced topics.* Cambridge, MA: MIT Press.

Hawkins, D. (1970). I, thou, it. In *The ESS Reader* (pp. 45-51). Newton, MA: Education Development Center.

Hillel, J., & Samurcay, R. (1985). *The definition and use of general procedures by nine year olds.* (Research Report No. 2). Montreal: Concordia University, Department of Mathematics.

Holmes Group. (1986). *Tomorrow's teachers.* East Lansing, MI: Author.

Hopkins, D. (1985). *A teachers' guide to classroom research.* Philadelphia, PA: Open University Press.

Hull, W. (1979). *Teachers' seminars on children's thinking: A progress report.* Grand Forks, ND: North Dakota Study Group on Evaluation, University of North Dakota.

Kemmis, S., & McTaggart, R. (1981). *The action research planner.* Victoria, Australia: Deakin University Press.

Kull, J. (1986). Learning and Logo. In P. Campbell & G. Fein (Eds.), *Young children and microcomputers.* New York: Prentice Hall.

Lawler, R. (1982). Designing computer-based microworlds. *Byte, 7*(8), 138-160.

Leron, U. (1983). Some problems in children's Logo learning. In R. Hershkowitz (Ed.), *Proceedings of the Seventh International Conference for the Psychology of Mathematics Education.* Israel: Weizmann Institute of Science, .

Loucks-Horsley, S., Harding, C., Arbuckle, M., Murray, L., Dubea, C., & Williams, M. (1987). *Continuing to learn: A guidebook for teacher development.* Oxford, OH: National Staff Development Council.

MacDougall, A. (1986). Children's difficulties in perceiving structure and using subprocedures. In C. Hoyles, R. Noss, & R. Sutherland (Eds.), *Proceedings of the Second International Conference for Logo and Mathematics Education* (pp. 30-37). London, UK: University of London, Institute of Education.

Martin, K., & Bearden, D. (1983). *The rule of 360.* Irving, TX: Martin-Bearden.

Olive, J. (1986). The collection and analysis of qualitative data in a Logo learning environment using dribble files. In *Proceedings of the 8th Annual Meeting of PME-NA.* East Lansing, MI: Michigan State University.

Papert, S. (1980). *Mindstorms.* New York: Basic Books.

Papert, S., DiSessa A., Watt, D., & Weir S. (1979). *Final report of the Bookline Logo Project, Part II.* (Logo Memo No. 53). Cambridge, MA: MIT Logo Group.

Pea, R., Kurland, M., & Hawkins, J. (1987). Logo and the development of thinking skills. In R. Pea & K. Sheingold (Eds.), *Mirrors of minds: Patterns of experience in educational computing* (pp. 178-197). Norwood, NJ: Ablex Publishing.

Polya, G. (1957). *How to solve it.* Princeton, NJ: Princeton University Press.

Schon, D. (1983). *The reflective practitioner.* New York: Basic Books.

Schon, D. (1987). *Educating the reflective practitioner.* San Francisco: Jossey-Bass Publishers.

Somekh, B. (Ed.). (1986). *Action research and the micro.* (CARN Special Bulletin). Norwich, UK: University of East Anglia, Classroom Action Research Network.

Thornburg, D. (1984). *Exploring Logo without a computer.* Palo Alto, CA: Addison-Wesley.

Thornburg, D. (1986). *Beyond turtle graphics.* Palo Alto, CA: Addison-Wesley.

Thyberg, L. (1987). The basis for a process approach to programming in Logo. In T. Lough & G. Bull (Eds.), *Proceedings of the East Coast Logo Conference* (pp. 183-186). Westport, CT: Meckler Publishing Corporation.

Turkle, Sherry (1984). *The second self.* New York: Simon and Shuster.

Watt, D. (1979a). *Final report of the Brookline Logo Project. Part III, Profiles of individual students' work.* (Logo Memo No. 54). Cambridge, MA: MIT Logo Group.

Watt, D. (1979b). A comparison of the problem-solving strategies of two students learning Logo. In D. Harris (Ed.), *Proceedings of the National Educational Computing Conference.* Iowa City, IA: University of Iowa.

Watt, D. (1983a). *Learning with Logo.* New York: Byte Books / McGraw-Hill.

Watt, D. (1983b). *Learning with Apple Logo.* New York: Byte Books / McGraw-Hill.

Watt, D. (1984). *Learning with Commodore Logo.* New York: Byte Books / McGraw-Hill.

Watt, D. (1985a). *Learning with IBM Logo.* New York: Byte Books / McGraw-Hill.

Watt, D. (1985b). Teacher-made microworlds, or training teachers to use Logo vs. training them to teach it. In J. Wilton (Ed.), *Proceedings of the Joint Conference of the Educational Computing Organization of Ontario, and the Association for Educational Data Systems* (pp. 143-146). Toronto: Association for Educational Data Systems.

Watt, D. (1988-1989). Assessing Logo learning in classrooms. (Series of nine articles). *Logo Exchange, 7*(1-9).

Watt, D. (1989, May). Research on Logo learning as a path to professional teacher development. *Electronic Learning,* pp. 22-24.

Watt, M. (1979). *Thinking about thinking: A description of a professional development seminar.* Brookline, MA: Teacher Center.

Watt, M. (1989a). A reflective teacher's interventions with a tentative learner of Logo: A case study. *The Computing Teacher, 16*(4), 51-55.

Watt, M. (1989b). When teacher and student are stumped. *The Computing Teacher, 16*(8), 30-32.

Watt, M. (1990). Logo teacher as Logo researcher. *The Computing Teacher, 17*(6), 27-29.

Watt, M., & Watt, D. (1982). Design criteria for collaborative classroom research. In T. Amabile & M. Stubbs (Eds.), *Psychological research in the classroom: Issues for teachers and researchers.* New York: Pergamon Press.

Watt, D., & Watt, M. (1986a). Starting Logo with young children? Slow down the turtle! *Logo Exchange, 5*(1), 3-5.

Watt, M., & Watt D. (1986b). *Teaching with Logo: Building blocks for learning.* Menlo Park, CA: Addison-Wesley.

Watt, M., & Watt, D. (1987a). Polishing programs for publication: A neglected step in the Logo programming process. *National Logo Exchange, 5*(9), 3-4.

Watt, M., & Watt, D. (1987b). Using the process approach to teaching writing as a model for teachers of Logo. In J. Hillel (Ed.), *Proceedings of the Third Conference on Logo and Mathematics Education (LME3)* (pp. 202-211). Montreal: Concordia University.

Watt, M., & Watt, D. (1988). Teachers as collaborative researchers: Professional development through assessing Logo learning. In W. Ryan (Ed.), *Proceedings of the National Educational Computing Conference, 1988* (pp. 62-71). Dallas, TX: ICCE.

Watt, D., & Watt, M. (1989). Rethinking Logo pedagogy using a process approach. In W. Ryan (Ed.), *Proceedings of the National Educational Computing Conference, 1989* (pp. 180-186). Boston, MA: ICCE.

Watt, D., & Watt, M. (1990, July). *How teachers improve their practices through action research: The case of the Logo Action Research Collaborative.* Paper presented at the World Conference on Computers in Education, Sydney, Australia.

Weir, S. (1987). *Cultivating minds: A Logo casebook.* New York: Harper and Row.

Weizenbaum, J. (1979). *Computer power and human reason: From judgement to calculation.* San Francisco: W. H. Freeman.

Appendix
Critical Aspects Checklist and Notepage

For an electronic copy of this list in a *Microsoft Word for the Macintosh* format, send a formatted 3.5" disk to Daniel Lynn Watt and Molly Lynn Watt, Logo Action Research Collaborative, Education Development Center, 55 Chapel Street, Newton, MA 02160.

Criteria for Critical Aspects

We define critical aspects of Logo learning to be those that:

- have been identified by teachers and researchers to be critical to students' effective use of Logo
- are not usually acquired by students without some form of timely teacher intervention
- are present (or available) in most Logo learning experiences
- are readily accessible to observation by classroom teachers.

Items on the checklist are intended to serve as guidelines for teachers to measure student skill levels. Other skills that teachers identify as they work with students using Logo can be added to the list as necessary.

Notes for Using the Checklist

1. The presence of a check or X on a line indicates the level of the student's mastery on a continuum. For example:
   X_____ means the student does this with specific encouragement and help
   _____X_____ means the student can use this skill or idea, but mastery is not complete
   _____X means the student has mastered this completely and can teach it to others.

2. Many of the critical aspects listed in Sections 1-4 can be observed by looking at printouts of a student's procedures. Most of the critical aspects in Sections 5-8 may be best observed by direct observation of student work, by looking at a dribble file, or by conversation with a student or teacher.

---

1. Procedures and Subprocedures

_____ A. Preprocedures

_____ B. REPEAT

_____ C. Sequential, undifferentiated procedures

_____ D. Use of subprocedures

_____ E. Structured programming styles

_____ F. Modular programming styles

_____ G. Procedures using recursion and conditionals

Teacher's Notes

The International Society for Technology in Education grants permission to photocopy this page for educational uses.

2. Procedures With Variables

_____ A. Prevariables

_____ B. Procedures with simple inputs

_____ C. Procedures with multiple inputs

_____ D. Using inputs with subprocedures

_____ E. Procedures with proportional variables

_____ F. Procedures that involve recursion, incrementing variables, and conditionals

_____ G. Using global variables (MAKE)

3. Mathematics of Turtle Geometry—Using 360 Degrees

_____ A. Understanding the relation: Rotation x number of sides = 360

_____ B. Distinguishing between complete and incomplete rotations

_____ C. Recognizing and using special angles (factors of 360 and their multiples)

_____ D. Using the total turtle trip in constructing figures

_____ E. Using state-transparent shape procedures as modules for larger projects

4. Mathematics of Turtle Geometry—Using Mathematical Thinking

_____ A. Estimating angles and distances

_____ B. Combining steps

_____ C. Using inverses:RIGHT/LEFT, FORWARD/BACK

_____ D. Using specific knowledge about particular shapes

_____ E. Using RIGHT/LEFT symmetry

_____ F. Using similarity

_____ G. Using transformational geometry

_____ H. Using and understanding wrapping

Teacher's Notes

The International Society for Technology in Education grants permission to photocopy this page for educational uses.

5. Learning Through Exploration

_____ A. Keeping records of exploratory work

_____ B. Taking advantage of accidents

_____ C. Using and looking for patterns

_____ D. Testing the limits

_____ E. Making and testing hunches and hypotheses

_____ F. Generalizing from one or more example

_____ G. Sharing ideas with others

6. Debugging Strategies

A. Identifying the bug

_____ A.1. Using Logo's error messages

_____ A.2. Describing the bug

_____ A.3. Categorizing and naming the bug

_____ A.4. Recognizing familiar bugs

B. Locating the bug

_____ B.1. Using Logo debugging aids

_____ B.2. Dividing a long procedure into parts

C. Fixing the bug

_____ C.1. Playing turtle

_____ C.2. Using mathematical knowledge

_____ C.3. Testing each part separately

_____ D. Incorporating the bug

_____ E. Reflecting on bugs to help eliminate future bugs

_____ F. Using planning strategies to help eliminate

Teacher's Notes

The International Society for Technology in Education grants permission to photocopy this page for educational uses.

potential bugs

7. Planning, Carrying Out, and Completing a Logo Project

A. Setting appropriate goals

_____ A.1. Using pictures and sketches as planning aids

_____ A.2. Dividing a problem into parts

_____ A.3. Using a superprocedure hierarchy and/or procedure tree in planning

B. Creating a working draft

_____ B.1. Simplifying the plan

_____ B.2. Setting specific preliminary and intermediate goals

_____ B.3. Beginning to write and test procedures

C. Revising and/or elaborating on a plan

_____ C.1. Eliminating goals that are too difficult

_____ C.2. Adding new goals or revising the entire plan as a result of new ideas

D. Sharing or publishing the results

_____ D.1. Debugging the final version

_____ D.2. Elaborating on the final version

_____ D.3. Sharing the work in a public form

8. Managing Workspace and Files

_____ A. Saving procedures on disk in an undifferentiated workspace

_____ B. Saving work in more than one file

_____ C. Distinguishing between file names and procedure names

_____ D. Using Logo commands to keep track of work

_____ E. Making journal notes or hard-copy printouts of workspace information

The International Society for Technology in Education grants permission to photocopy this page for educational uses.

LONG DISTANCE LOGO

Educators–You don't have to go to classes to earn graduate credit–let the classes come to you! ***Introduction to Logo For Educators***, a graduate level ISTE *Independent Study* course, allows you to learn at your own pace while corresponding with your instructor by mail, fax, or telecommunications. This course is available for *LogoWriter* and Terrapin's *Logo PLUS*.

WORK INDIVIDUALLY OR WITH A GROUP

Take ***Introduction to Logo For Educators*** at home, or study with a group of colleagues. The course uses video tapes (ON LOGO) with MIT's Seymour Papert, printed materials, textbooks, and disks. View the tapes, read and report on course materials, do projects, design Logo lessons for students, and correspond with your instructor by mail, fax, or telecommunications.

NOT JUST ANOTHER CLASS

Dr. Sharon Yoder, editor of the *Logo Exchange*, designed ***Introduction to Logo For Educators*** to provide staff development and leadership training. The four quarter-hour course meets the standards of the College of Education at the University of Oregon, and carries graduate credit from the Oregon State System of Higher Education.

ON LOGO VIDEO TAPES

School Districts may acquire a license for the use of the ON LOGO package of 8 half-hour videotapes and 240 pages of supporting print. For a one-time fee, the package may be obtained with both tape and print duplicating rights, enabling districts to build libraries at multiple sites.

Group Enrollment. Reduced tuition is available to institutions that enroll a group of six or more educators. This special price does not include the ON LOGO videotapes. Your group must acquire the tapes or have access to them. Once acquired, the library of tapes and materials may be used with a new groups enrolling for the same reduced fee.

Individual Enrollment. Educators with access to the tapes may enroll individually. A materials fee is charged for texts and a packet of articles. This fee is waived for enrollees who already have the texts.

To receive an Independent Study course catalog, write or call:

International Society for Technology in Education
Independent Study Department
1787 Agate Street, Eugene, OR 97403-1923
Phone: 503/346-2412 Fax: 503/346-5890
Internet: ISTE@Oregon.uoregon.edu

ISTE BRINGS THE WORLD OF TECHNOLOGY CLOSER TO YOU.

By drawing from the resources of committed professionals worldwide, ISTE provides support that helps educators like yourself prepare for the future of education.

ISTE members benefit from the wide variety of publications, specialized courseware, and professional organizations available to them.

They also can enjoy exciting conferences, global peer networking, and mind-expanding independent study courses.

So if you're interested in the education of tomorrow, call us today.



International Society for Technology in Education
1787 Agate Street, Eugene, OR 97403-1923
Phone: 503/346-4414 Fax: 503/346-5890
Order Desk: 800/336-5191
CompuServe: 70014,2117
Internet: ISTE@Oregon.uoregon.edu
ISTE.Net (GTE): ISTE.office

AL SOCIETY FOR

WE'LL PUT YOU IN TOUCH WITH THE WORLD.