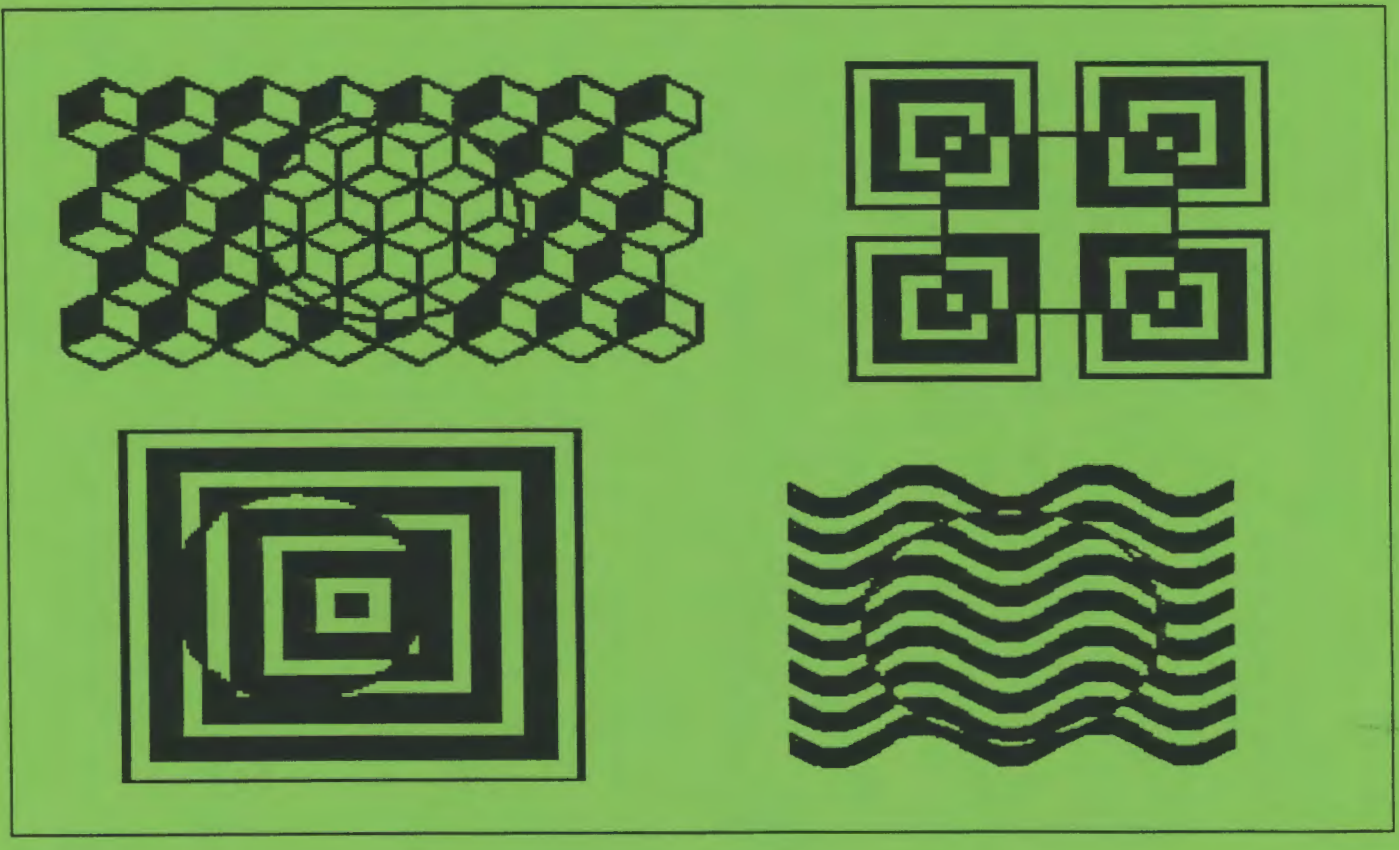
Journal of the ISTE Special Interest Group for Logo-Using Educators

LOGO EXCHANGE

February 1991 Volume 9 Number 5



International Society for Technology in Education

ISTE Publications

Make computer education a win/win situation in your classroom.

Research indicates inequities exist in nearly all classrooms. *Yes I Can: Action Projects to Resolve Equity Issues in Educational Computing* clearly explains these issues and details actions to achieve positive results. The projects target the problems of females, minorities, and kids with special needs, whose access to computer labs is often limited.

Drawn from the results of several action projects by the Educational Computer Consortium of Ohio (ECCO), *Yes, I Can* shows the way to:

- improve student's self-esteem
- use a word processor as a motivational aid
- emulate role models' successes
- improve administrators' understanding of the role of computers in a educational setting

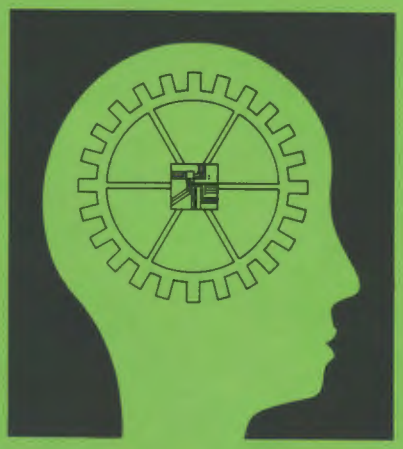
ISTE'S *Yes I Can* also details how you can get other teachers involved in computer equity. References for projects are included with the book.

Yes I Can: Action Projects to Resolve Equity Issues in Educational Computing $15.00 plus $3.25 shipping. Contact ISTE, 1787 Agate St., Eugene, OR 97403-9905; ph. 503/346-4414.

The world is full of inequity, but when it comes to changing it in your class—yes you can!

Fantastic Journey Through Minds and Machines

by Michael Muir



Artificial Intelligence (AI) calls up visions of supercomputers and gigabytes of computer code. Michael Muir gives AI a human dimension and a practical application in the secondary classroom.

Even noncomputing readers will enjoy his entertaining chapter introductions on the important history and people of this technical and intellectual frontier. Muir explains the essential challenges of creating robots, games, natural language, machine intelligence, and expert systems.

For students and teachers of computer science, each chapter also includes a complete AI program in Logo (both *LogoWriter* and *Logo Plus* versions). The programs range in sophistication from a simple maze to an inference engine with natural language interface. The code is documented line by line, so that even novice programmers can follow the logic and development. Suggestions for program extensions offer expert-level challenges.

Fantastic Journey Through Minds and Machines is $13.95 plus $3.25 shipping. Contact ISTE, 1787 Agate St., Eugene, OR 97403-1923; ph. 503/346-4414.

LOGO EXCHANGE

Volume 9 Number 5 Journal of the ISTE Special Interest Group for Logo-Using Educators February 1991

Logo Exchange is the journal of the International Society for Technology in Education Special Interest Group for Logo-using Educators (SIGLogo), published monthly September through May by ISTE, 1787 Agate Street, Eugene, OR 97403-1923, USA; 503/346-4414. This publication was produced using Aldus *PageMaker*®.

POSTMASTER: Send address changes to *Logo Exchange*, ISTE, 1787 Agate St., Eugene, OR 97403-1923. Second-class postage paid at Eugene OR. USPS #000-554.

ISTE is a nonprofit organization with its main offices housed at the University of Oregon.

ContentsContents

ISTE Membership

	U.S.	Non-U.S.
	36.00	43.00

SIGLogo Membership (includes *The Logo Exchange*)

	U.S.	Non-U.S.
ISTE Member Price	25.00	30.00
Non-ISTE Member Price	30.00	35.00

Send membership dues to ISTE. Add $2.50 for processing if payment does not accompany your dues. VISA, Mastercard and Discover accepted. Add $18.00 for airmail shipping.

From the Editor

Visions

As I was editing this issue of *LX*, I was particularly struck by Judi Harris' "Logo LinX" column. In it she describes a particular scene from a NOVA program called *Talking Turtle*. Perhaps you have seen it? This program intersperses interviews with Seymour Papert and a variety of Logo teachers from all over the world with scenes of children of all ages using Logo. It is a beautiful work providing a wondrous vision of Logo and the Logo philosophy.

This program is a special favorite of mine. I used to show it to my high school students just as they were beginning turtle graphics. Besides showing them that kids of all ages could write Logo programs more complex than they were yet able do, it gave them a vision of what Logo was all about. Now *Talking Turtle* provides a basis for thoughtful discussions among my graduate students.

The particular scene Judi describes has always been one of my favorite moments in the program. It shows two young girls who have created a Logo-like description of a dance which they then perform. Perhaps I have loved that scene because I have a daughter who has studied dance seriously for over 10 years. Perhaps my attraction was because I myself love music and dance. Perhaps my affection runs deeper, however. That particular scene shows a wondrous combination of technology and the arts. There is a depth of ownership of Logo ideas in those girls that allows them to use Logo in everything they do. When I first saw *Talking Turtle*, using Logo to create a dance was, for me, a unique "extension" of Logo.

Today we see an ever-increasing number of extensions to Logo. In this issue *LX*, there a several articles on the use LEGO blocks to build models controlled by Logo. In earlier issues this year, we have seen Logo extended into telecommunications by *LogoExpress*. Each month Glen and Gina Bull show us yet another way to connect Logo to the outside world through the emerging world of multimedia.

Since *Talking Turtle* was made, Logo implementations have changed dramatically. These newer versions are more powerful and allow a great deal more initial success and excitement. Our students used to draw their first square and then say "How do I fill it with color?" or "How do I put a label on it?" These once difficult tasks are now almost trivial with *LogoWriter* and *Logo PLUS*. More recently, we have seen an upsurge in interest in Logos for the Macintosh. While *Terrapin Logo for the Macintosh* has been available for quite a few years, many people are just now discovering its ability to work with multiple turtles and a variety of turtle shapes. *LogoWriter* for the Mactintosh offers a number of exciting new capabilities not available on IBM or Apple computers.

More recently, *Object Logo* has reappeared in a new package from Paradigm Software. (See the announcement elsewhere in this issue.) This version has two to three times as many primitives as any other version of Logo on the market. It allows extensive access to the capabilities of the Macintosh and extends the "ceiling" of Logo even more by providing powerful object-oriented programming capabilities. Paradigm is committed to extending the capabilities of *Object Logo* even further (and they deserve our support so that they can do so).

So where will we go next? What will the next threshold look like? One of our graduate students here at the University of Oregon, Victor Chen, is working on a vision of the future for his master's project. Victor envisions software that would provide "building blocks" for a teacher to create customized software for her students without needing to be a sophisticated programmer. He sees a "point and click" environment that would allow, for example, putting together a simulation, low-level word processor, medium-level spread sheet, and simple command-language module (Logo?) to fit the needs of a particular student or particular assignment. One program that touches the edge of his vision is the *Video Works* program, but there are elements of his ideas in programs like Microsoft *Word* that allow you to customize user levels and menus, or *HyperCard* which allows easy connections among applications. Victor's ideas require interactions between many different types of software through a complex and intelligent operating system. We certainly can't implement Victor's ideas on today's school computers, but then look how far we have come in the last 10 years. We can certainly dream—let us never stop dreaming.

> *Little by little, we are learning how the computer can make bridges between the mathematical and the sensual; the abstract and the intimate.*
> —Seymour Papert in *Talking Turtle*

Sharon Yoder
ISTE
1787 Agate Street
Eugene, Oregon 97403
Ph: 503-346-2190
CIS: 73007,1645
BITNET: YODER@OREGON

Monthly Musing

Stick Your Neck Out
by Tom Lough

As many of you do, I read a wide variety of magazines. I frequently encounter some delightful surprises.

I was thumbing through a television trade magazine called *Channels* recently, when I happened upon a full page notice (I really couldn't call it an ad) featuring 10 brief sentences. And right in the middle of the text was a beautiful drawing of a turtle. Needless to say, it caught my attention!

The message was straightforward and simple. It was signed by Bill Daniels, a cable television broker operating out of Denver. His message spoke to the teacher in me. Here is what he said:

> It has made millionaires out of paupers. It has earned the Nobel Prize. And by using it wisely, entrepreneurs have ended up running corporations. I'm referring to believing in yourself and your ideas. Taking a chance on your own merit and giving the world your best shot. Somebody once said, "Observe the turtle, he progresses only with his neck out." (The turtle drawing appeared here.) I think the same holds true for us two-legged creatures. If you've never taken a chance on yourself, at least think about it. Unless you do, you may never know what you're missing out on. And neither will the rest of the world.

Thank you very much, Mr. Daniels, for your inspiring message.

OK, folks, there it is. Something for each of us and for our students.

Some psychologists suggest that we all operate within what is called a "comfort zone." We are competent in this comfort zone; we have a routine and know exactly what to do. Outside that zone things are not quite so predictable, so we often hesitate to venture out.

The Logo turtle has served as many a metaphor. Now it is inviting you to take a risk, to stick out your neck and take a chance on yourself.

This risk-taking spirit is at the very heart of Logo. Look at the attitudes of your students as they puzzle their way through to the successful completion of a project. Sometimes they didn't know where they were heading when they started, and yet they finished up with something surprisingly respectable. Sometimes they had worked out a detailed plan in advance, only to find that it had to be changed in major ways later. But they took their chances and gave it their best shot.

What about you? Do you have a dynamite idea for a new class activity or a new direction for curriculum development? Have you dreamed of writing an article for *LX* about that wonderful fifth-period class Logo project? Are you thinking about trying a completely different way of teaching recursion this time around?

Come on! Give it a try! Don't you owe that much to yourself and to your students?

FD 100!

P.S. I was pleased to receive several encouraging reactions to my October lament about Logo and secondary schools. One response in particular might be of interest to *LX* readers. Over the past five years, Ron Place and his fellow teachers have developed and refined a high school Logo curriculum. I was pleased to hear that the use of this curriculum in Ron's high school has provided an excellent introduction to Pascal. The curriculum has been published in three-ring binder form for use with *Terrapin Logo* and *Logo PLUS*. For more information, write to Ron Place, Logo Curriculum Publishers, 4122 Edwinstowe Avenue, Colorado Springs, CO 80907.

Tom Lough
Founding Editor
PO Box 394
Simsbury, CT 06070

About the cover

Orlando Mihich, Technology Coordinator for his school in Manhattan, writes that he has done many curriculum-integration projects in his school including projects in architecture, physics, biology, AIDS, drugs, and substance abuse. This month's cover is from some "playful activities" that included work in Afro art, drawings from the Simpsons, and these beautiful drawings, which he calls Optical Art. An article about Orlando's work with his students appeared in the October 1990 issue of *The Computing Teacher*. He can be reached at 339 Pacific Avenue, Jersey City, NJ 07304.

Logo Ideas

What's Your Sound?
by Eadie Adamson

In November 1988 "Logo Ideas" I discussed working with music (see "A Logo Concert," pp. 6 - 8, *Logo Exchange*, November 1988). Since that time there have been some interesting extensions of the use of music with *LogoWriter*, one of which will interest the "non-musicians" in the audience.

How It All Started

It all began last October after I returned from giving a workshop for the St. Paul Logo Project. I had promised my students I would share with them what the teachers in St. Paul created during the workshop. I brought back a wonderful multi-page story about a spider, a witch and a cat, rhymed and set to the music of "Itsy Bitsy Spider." The project was complete with text and animation as well as sound. While many of my students, having seen this, then wanted to learn about animation, one group was particularly interested in making music. Naturally I was ready to jump in. We immediately began to explore *LogoWriter's* **tone** primitive.

Testing Tone's Inputs

The **tone** primitive in *LogoWriter* takes two inputs. First we tested these two inputs required by **tone**, trying to find out what each did. A period of experimenting followed. It took a while before the group agreed that the *kind* of sound was determined by the first input, which represents the frequency of the sound, and that the *length* of the sound, the duration, was determined by the second input.

Next we explored the maximum and minimum effective numbers for each input. Error messages appeared for inputs which were too large: **tone doesn't like 9999** as input, for example, when someone had typed **tone 300 9999**. This experience in itself was good for beginners who often ignore error messages. Paying attention to the error message made clear which input was unacceptable. Students had an excellent opportunity to practice reading and interpreting error messages during this experiment. The class shared the work on the project, quickly refined strategies, and then shared their results.

Class time for that day was exhausted by then. Next time: procedures!

Introduction to Procedures

This group of students had not yet learned about the power of writing procedures. Here was an opportunity to teach an important Logo programming skill in a context that was meaningful to the children—an essential component of good learning and the ideal Logo environment! Since my students had already made shapes and learned to copy and paste shapes, they had some useful prior knowledge with which to begin learning about procedures. I simply suggested that, just as there was a flip side to the shapes page, there was a flip side to a page. They already knew how to flip back and forth on the shapes page. Immediately everyone flipped to the "flip side."

We started with middle C procedure, wrote a procedure like this:

```
to c
tone 273 10
end
```

(Note: My frequencies are about a half-tone different from the ones in the *LogoWriter* manual. I got them from a musician, Jim Wingate, who worked them out with an oscilloscope. You can use mine or LCSI's. Musicians with perfect pitch will probably be happier with mine. See below for the full list of frequencies.)

We wrote a few more procedures following the same pattern.

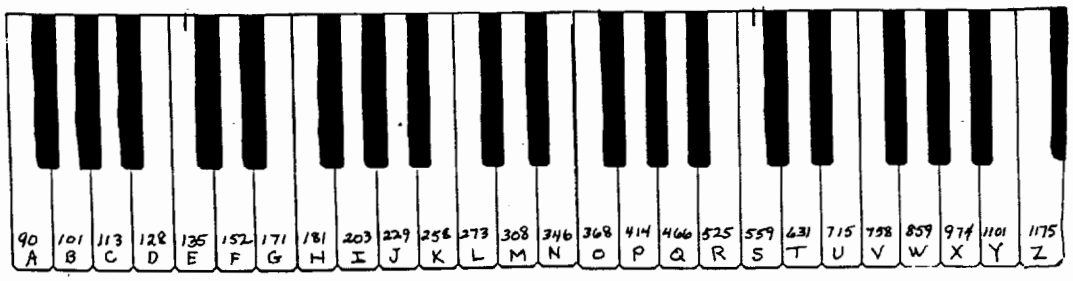
A New Idea for Sound

Suddenly, Jeffrey said, "What about making a note for every letter of the alphabet so we can play our names?" Great idea! Again, time was up, but I promised the next day to supply a frequency for each letter of the alphabet. Secretly, I had found a wonderful way to teach these students to **cut** and **paste** text too: write one procedure, **copy** it, **paste** it, change the letter and the frequency. It would be easy for them to build an alphabet in far less than a class period!

I made a copy of a piano keyboard and put the frequencies of the white notes on the keys, along with the letters of the alphabet. (See the top of the next page.) For the musically inclined, this clear the relationship between the sounds and the frequencies.

Writing the Alphabet Procedures

To begin this lesson, everyone wrote the procedure for "A." Together we all put our cursors on the first letter of the procedure (the "t" in "to"). Together we pressed the Open Apple key, held it down and pressed the "1" key. Then we all pressed the right arrow keys until our whole procedure was highlighted. I took a few seconds to be sure no one was in trouble.



Now we all pressed Open Apple key and the "3" key. The white highlighting disappeared, of course. Everyone moved their cursor below the first procedure and pressed the Return key to skip a line. Together we all held down the Open Apple key while pressing "4." Voila! Another copy of the first procedure. Time to write procedure **B**, but now all we had to do was change the **A** in the first line to **B**, creating a procedure for **B**. Then we needed to change the frequency (the first input to tone) to the frequency for **B** from the reference page that I gave them. I used my demonstration monitor to show the copying and pasting sequence where everyone could refer to it when necessary. In no time, whole alphabets were ready.

```
PUT THE CURSOR ON THE T IN "TO"
     PRESS APPLE AND 1
ARROW KEYS TO HIGHLIGHT
     APPLE AND 3 TO COPY
MOVE TO WHERE YOU WANT TO PASTE
     APPLE AND 4 TO PASTE
```

The SOUND of a Name

Ivana was careful to write the procedures for just the letters of her name first. Suddenly she called out excitedly, "Listen to my name!" It sounded a little like R2D2. Soon everyone had their names playing, or almost ready to play. We had a room full of delightfully funny electronic sounds.

But there's more...

I began thinking about the difficulties of remembering to put a space between each letter. If we showed their teacher or a friend from another class what we were doing, I thought we would want something that was easier to use and, for the uninitiated, a little more mysterious in its operation. If someone simply typed their name, Logo would respond with an error message. We needed a way around that!

While the class was working, I dashed off a procedure which took a word as input and would play it a note as a time:

```
to play :name
if empty? :name [stop]
run parse first :name
play butfirst :name
end
```

By itself this is good, but the name (the input for **play**) needs to be quoted for the process to work. We really would not find that much more helpful. Obviously (to me) we could write a little program which asked for a person's name, then played it. Here it is:

```
to music
type [What's your name?]
type char 13
play (first readlistcc "name)
end
```

More Extensions

The idea of a talking computer amused me. I thought of the R2D2-like sounds we had already heard. I created a "**talk**" procedure which accepts a list as input. The list is a whole sentence. **Talk** plays each word, pausing briefly before playing the next:

```
to talk :list
if empty? :list [stop]
play first :list
wait 20
talk butfirst :list
end
```

What about punctuation? You can write procedures for commas, periods, question marks, even exclamation points. They should be different frequencies from the notes. One might simply differentiate a comma from a period by the duration of the sound: short for a comma, longer for a period.

Then the same frequency would indicate punctuation. Here are some samples:

```
to .
tone 37 10
end

to ,
tone 37 5
end
```

A question might have a high tone; an exclamation point still higher. This could be fun!

Musical sounds treated this way could lead to writing whole computer conversations, or making up codes (can you really remember all those frequencies?). Can you learn the sound of a word? IT might be fun to try! What does "LOVE" sound like?

Put the Words on the Screen as They Play

Kylie wondered about putting the letters on the screen as they played. This can be done simply by adding a line to each procedure to **insert** the text on the screen. The primitive **insert** puts text on the screen and leaves the cursor on the line. A series of **insert** commands could put together a word. Again, using **copy** and **paste** makes the task of adding this line to each procedure a little easier. Here is how the procedure for "a" would look:

```
to a
tone 90 10
insert "A                    <-Words or names
                                will print in caps
end
```

Another possibility would be to adapt the **play** procedure so that it inserted the word it played. One could do this instead of changing each letter:

```
to play :name
if empty? :name [stop]
run parse first :name
insert first :name    <-This line is added
play butfirst :name
end
```

Play will then insert the first letter after it plays, leaving the cursor to the right of the letter. After the last letter is inserted, the cursor will remain at the end of the word. Then the **talk** procedure could insert a space after each word: **insert char 32**. **Talk** could end by printing an empty list, **print []**, to move to the next line. The adjusted procedure might look like this:

```
to talk :list
if empty? :list [print [] stop]
play first :list
wait 20
insert char 32 <-This line is added

talk butfirst :list
end
```

Thinking About the Sound of Language

This project could become immensely complicated. One might try to distinguish between vowels and consonants by making the vowels have slightly longer duration than the consonants. Then it might be possible to begin to identify words by their rhythm as well as melody. This kind of project could lead students into a discussion of the origins of language, a look at linguistics from a new angle, and even a study of the "rules" of pronunciation and the rhythm of speech.

Wonderful musical compositions might be constructed with the musical alphabet. How about writing a fugue based on the sound of your name? Can you play names in counterpoint using several computers? What about a musical sentence as the theme for a song? Will a young musician, using this idea to explore melody, come up with the first alphabetical symphony?

For Those with GS or MS DOS LogoWriter

Consider giving each letter a color. There are not enough colors to have a text color for each letter, but how about having some colors produce the letter in color while others cause the screen to flash. How many times would the screen flash—and in what color or colors—for an exclamation point? a period? You would need to make these changes in the procedures for the individual letters.

The Frequencies We Used

On the next page is a list of the frequencies we use for music (using the # for the sharp, B flat is then A#):

LLF	90	MIDDLE C	273	HG	859
LLF#	95	C#	290	HG#	913
LLG	101	D	308	HA	974
LLG#	107	D#	326	HA#	1034
LLA	113	E	346	HB	1101
LLA#	120	F	368	HHC	1175
LLB	128	F#	391		
LC	135	G	414		
LC#	143	G#	439		
LD	152	A	466		
LD#	161	A#	495		
LE	171	B	525		

LF	181	HC	559
LF#	192	HC#	593
LG	203	HD	631
LG#	216	HD#	669
LA	229	HE	715
LA#	243	HF	758
LB	258	HF#	806

Special thanks to my students who were the inspiration for this article: Jeffrey, Ivana, Kylie, Adam, David, Elizabeth, Elliot, Melanie, Russel and Nadia. Thanks also to the Logo St. Paul teachers who were the creators of the "The Itsy Bitsy Spider" program: Carol McCarty, Clare Eldredge, Nancy Kubat, Caroline Nicholas, Diane Hankes, Grace Bellesen, and Mary Tacheny.

Eadie Adamson is a member of The New Laboratory for Teaching and Learning at The Dalton School in New York, where she works with Logo and LEGO/Logo with Middle School students.

Eadie can be reached via *LogoExpress* on the New York host, 1-212-765-4924 (from Los Angeles, the number is 1-818-505-1511). Eadie's username is EadieA.

Eadie Adamson
The Dalton School
108 East 89th Street
New York, NY 10128

LME5 Conference
April 1 - 5, 1991
Cairns, Australia

LME5 is a conference for those interested in mathematics education research in which computers play a significant part in providing the medium for expression of the mathematics.

The themes chosen for LME5 are:

- Logo in the mathematics curriculum;
- styles of learning and strategies for teaching mathematics in a programming environment;
- expressing mathematical structures in programming languages.

Dr. Richard Noss (Institute of Education at London University) will be the program chair. Papers presented will be published by ACER and available at LME5.

Timelines
Abstracts of papers will be expected in Australia at ACER by 30 November 1990 and final papers by 31 January 1991.

Registrations will be accepted in the order in which they come, but places will be reserved for those who have attended LME on a regular basis in the past.

Location
LME5 will be held at Lake Tinaroo in the hills just outside Cairns, a major tourist resort on the northern Australian coast beside the beautiful Great Barrier Reef. The venue is one hour's drive from the coast.

As Australia is very far from home for many people, we are keen to make the voyage worthwhile. We are planning to confer from April 1 - 5 and then travel with our visitors to the Great Barrier Reef for two days of diving and snorkelling. We would like overseas visitors to travel south to Sydney with us for a half-day mathematics education plenary session with teachers on Monday April 8 as a way of gaining an opportunity to see Sydney.

Send inquiries to:

LME5 c/o Liddy Nevile
A.C.E.R.
P.O. Box 210
Hawthorn
Victoria 3122
Australia

Questions Please!

by Frank Corely

This month's column comes to you courtesy of Brian Harvey. Consider this column editor just a vehicle to communicate his thoughts. I take blame for any mistakes in the column, but he receives full credit for anything which is correct. Brian was gracious enough to respond at great length to the October *Questions, Please* column. This month's column consists essentially of his answers to some key questions in that column. Thank you, Brian Harvey.

2. Please explain in simple language the LPUT command, how it works and how it is used.

Let's suppose you want to write a procedure FROMT1TO that takes a number as input and outputs a list of numbers from 1 to that input. For example, SHOW FROM1TO 10 should print [1 2 3 4 5 6 7 8 9 10]. The most natural way to express this is:

```
TO FROM1TO :NUMBER
IF EQUALP :NUMBER 0 [OUTPUT []]
OUTPUT LPUT :NUMBER FROM1TO
   :NUMBER - 1
END
```

In the second instruction line we want to append one number (:NUMBER) at the right end of a list that was produced by the recursive invocation. It's easy to imagine other situations in which you'd want to add something at the right end of a list.

```
TO POLITE :ORDER
OUTPUT LPUT "PLEASE. :ORDER
END

PRINT POLITE [BE QUIET]
BE QUIET PLEASE.
```

In both of these examples, just as is often the case with FPUT, we could use SENTENCE instead, in this way:

```
TO POLITE :ORDER
OUTPUT SENTENCE :ORDER "PLEASE
END
```

The use of FPUT and LPUT instead of SENTENCE is important when dealing with lists that contain sublists, like [VANILLA [RUM RAISIN] CHOCOLATE], which is a list of three elements, one of which is itself a list. We could add a single-word flavor like GINGER to the list either way:

```
PRINT SENTENCE [VANILLA [RUM RAISIN]
   CHOCOLATE] "GINGER
```

or

```
PRINT LPUT "GINGER [VANILLA [RUM
   RAISIN] CHOCOLATE]
```

but if we want to add a multi-word flavor we must say

```
PRINT LPUT [FUDGE SWIRL] [VANILLA
   [RUM RAISIN] CHOCOLATE]
```

not

```
PRINT SENTENCE [VANILLA [RUM RAISIN]
   CHOCOLATE] [FUDGE SWIRL]
```

5. In what sequence should the commands and concepts of Logo be taught to introduce programming in the language?

6. Are there some activities which are particularly well-suited for beginners to the language: both beginning teachers and teachers teaching to beginning students?

For both of these questions, there is no single right answer, but there are many books you can consult that show possible approaches. With young students, most teachers start with turtle graphics. Teaching high school or college students, I prefer to avoid graphics altogether, partly because some students think that it is "baby stuff" and partly because list processing emphasizes operations (procedures that output a value) over commands (procedures that just do something on the screen). This shift in emphasis is important to lead into the study of more advanced ideas in computer science later.

Two classic introductory books that provide both sequence and activity ideas are Harold Abelson's *Apple Logo* and Daniel Watt's *Learning with Logo*, both published by McGraw-Hill. For junior high school students I recommend *The Logo Project Book* by Alison Birch, published by Terrapin. For high school and college students, and for teachers learning Logo, I like my own *Computer Science Logo Style* series, published by MIT Press. There are, of course, many more books I could mention, but I'm trying to provide a short starting point rather than a comprehensive bibliography.

Another kind of answer is to think of Logo programming not as a topic in itself but as a tool in presenting other kinds of

curricula. MIT Press publishes a series called Explorations in Logo with books that use Logo to explore linguistics, visual modeling, and some topics in mathematics. These books are intended primarily for high school students.

7. Can someone give a typical page of a Logo lesson plan, including goals and objectives?

These days a lot of Logo people are willing to entertain questions like this, and you'll probably get some answers that look like what you wanted. But I'm an old-fashioned Logo person, and I think that the whole point of Logo is to get away from planned lessons. My idea of a perfect secondary computer class is that the teacher walks into the room, the kids are already at work, and six kids jump at the teacher to ask urgent questions. If the answer could be looked up the teacher just says "look it up!" In the cases where a kid really is having trouble understanding something, and has already asked other kids without success, the teacher goes over and teaches that kid while the other kids happily work on their own. About 10 minutes into the period, the teacher next door comes over to complain about the noise, and the computer teacher tells the other teacher to go jump in a lake.

If we're talking about elementary school kids who are in the same classroom all day, then there are a lot of computers in a corner of the room, and kids are constantly coming and going as something comes up in their other work that seems to suggest using a computer.

Of course this wonderful state of affairs presupposes that the kids know something about Logo programming already; their first days with the computer can't be quite so loose. Still, most of the planned lessons I've seen strike me as deadly boring. You can find whole books full of Logo worksheets, designed to make computer programming feel just like spelling or multiplication tables, but I don't recommend them.

8. Are there Logo institutes around the country for teachers and supervisors just beginning in Logo?

I'll mention the one that I'm involved with. There is an NSF-sponsored summer program at Kent State University called the Institute for Secondary Mathematics and Computer Science (IFSMACSE). This is not a "Logo institute," in that Logo is not the central concern, but Logo is heavily used throughout the curriculum of the institute. There are three separate strands: computer science, new topics in mathematics, and the use of computers in learning mathematics. I teach in the first of these strands. For more information, write to the College of Continuing Studies, Kent State University, Kent, OH 44242.

Once again, my great gratitude to Brian Harvey for his quick and thorough responses to these questions. I hope his answers have responded to the issues, both large and small, in a helpful and thought-provoking manner. Until next month, I hope that all of your questions are answered, but if they aren't, send them to me!

Frank J. Corley
St. Louis Priory School
500 South Mason Road
St. Louis MO 63141

CALL FOR PAPERS

Submission deadline: Feb. 28,1991

Educational Use of Computer Based Media in the Information Society

September 13-19, 1991
Tokyo, Japan

Co-sponsored by:
Japan Association
for Educational Technology &
International Society for
Technology in Education

The use of multimedia in education and training will be the focus of this conference. The conference will include both an English and Japanese strand. Prospective authors should submit two copies of a one-page abstract.

To submit abstracts or to receive more information, contact:

ISTE, % Multimedia Conference (Japan),
1787 Agate St.
Eugene, OR 97403-1923
ph. 503/346-4414 FAX: 503/346-5890

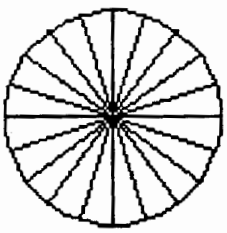
Beginner's Corner

Free-Wheeling in Logo
by Dorothy Fitch

One of the exciting things about Logo is the opportunity it gives us to open up a window into people's minds and watch them think. It is fascinating to watch people solve a problem, and even more interesting to observe several people solving the same problem—generally in quite different ways.

This month we'll look at a problem and think about several approaches to solving it. In the course of this, we'll be exploring different ways of creating circles in Logo.

Here's the problem:



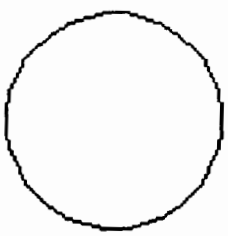
How would you go about drawing this bicycle wheel?

You don't actually need a computer for this exercise. The point is not necessarily to write a program to draw it, but just to think about *how* you would draw it. We *will* be writing some Logo programs here, however, so you won't be left with questions about how to do it!

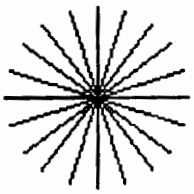
Here are some off-computer questions to ponder.

The first thing to think about is what building blocks would be useful? What parts make up the wheel?

If you had a circle,



you could add spokes to it.



Or you might want to use a pie-shaped piece as a building block.

We'll focus on different ways of drawing the wheel using the circle and spokes. The concept of using a pie-shaped piece sounds much easier to do than it actually is. If you are up for an interesting mathematical challenge, go ahead and try it, but we'll stick to circles and spokes!

Just for fun, draw the wheel with pencil and paper. Is this a good way to draw the wheel in Logo? Not really. You probably drew a circle, then the spokes from one edge of the circle to the other. You drew a vertical spoke, then a horizontal spoke, then the ones in between. The wheel above has 20 spokes, but that's a hard number to draw free-hand. A wheel with 8, 12, 16, or 24 spokes is easier.

So how might you draw the wheel in Logo?

Let's start with the spokes You may have drawn other Logo designs that are similar. Draw on paper how you want the turtle to move. Think of how you could use a REPEAT statement. Here's one solution:

```
TO  SPOKES
REPEAT  20  [FORWARD  40  BACK  40
   RIGHT  18]
END
```

The FORWARD and BACK distance determines the radius of the circle. This procedure draws 20 spokes. If you want a different number of spokes, use a different number after REPEAT and adjust the RIGHT turn number accordingly. (Remember that the Total Turtle Trip Theorem tells us that for the turtle to return to its original heading after completing a design, it will have turned a total of 360 degrees.)

Now for the circle. There are a couple of ways to think about a circle. A math teacher will tell you that a circle is a set of points equidistant from a central point. But most people using Logo think of a circle as a shape with many sides—the more sides a shape has, the rounder it looks!

Let's draw a Logo circle using each of these methods. Then you can decide which makes more sense to you. Either is correct.

Here's a procedure that draws a circle based on the "math teacher" definition: a set of points the same distance from the turtle's position.

```
TO CIRCLE
REPEAT 360 [PENUP FORWARD 40 PENDOWN
   BACK 1 PENUP BACK 39 RIGHT 1]
PENDOWN
END
```

If you type SPOKES and CIRCLE (in either order), you get a wheel! It takes a while to draw it, as we are making the turtle cover a lot of territory. (To speed it up, type HIDETURTLE before drawing the wheel. Now the computer doesn't have to spend a lot of time showing the image of the turtle as it zips around the screen.)

So, your completed procedure might look like this:

```
TO WHEEL1
HIDETURTLE
CIRCLE
SPOKES
SHOWTURTLE
END
```

This method is fairly straightforward. It is also a good approach for younger children who may be able to understand the concept of drawing a circle in this way.

Now let's try the many-sided shape approach. We can approximate a circle by having the turtle move forward a little, turn a little, move forward a little, turn a little, and so on, until it reaches its original position, like this:

```
REPEAT 36 [FORWARD 5 RIGHT 10]
```

That looks like a circle—by the time your shape has more than 20 sides, it looks pretty round—but it is not big enough for our spokes. We'll have to make it the right size. We can either spend time using trial and error, which isn't a bad approach, or use some math. (You can experiment with trial and error on your own.)

To get the circle the right size, we need to know the formula for the circumference of a circle, which we recall is $2\pi R$. R is the radius of the circle, 40 in this example, and π is roughly 3.14159 (close enough for our purposes).

You might want to define a procedure called PI. (Terrapin Logo for the Macintosh already has a built-in PI command.)

```
TO PI
OUTPUT 3.14159
END
```

You can get Logo to tell you the circumference of the circle by typing:

```
PRINT 2 * PI * 40
251.327
```

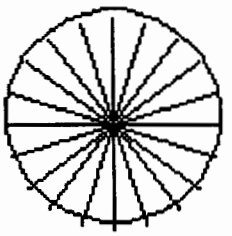
This is the total distance that the turtle should go forward. If it is moving forward 36 times, then each time it moves it should travel 251.327÷36 turtle steps. This number works out to be 6.98, which we can round to 7 with no visible negative effect (at least on my Apple screen). We can write a procedure called CIRCLE2 that looks like this:

```
TO CIRCLE2
REPEAT 36 [FORWARD 7 RIGHT 10]
END
```

Test it by typing CIRCLE2. Then move the turtle to the center of the circle using these instructions:

```
PENUP RIGHT 90 FORWARD 40 LEFT 90
   PENDOWN
```

Now type SPOKES to complete the wheel.



What happened? The spokes didn't fit quite right! That's because in our CIRCLE2 procedure, the turtle moves forward 7 steps before it turns the first time, causing the center of the circle to be shifted up the screen a little from where you expected it to be.

We can fix this by moving the turtle back half this distance before it begins the circle and moving it forward the same amount when it is done, like this:

```
TO CIRCLE2
BACK 3.5
REPEAT 36 [FORWARD 7 RIGHT 10]
FORWARD 3.5
END
```

Here's a completed version of WHEEL2, using the steps described above.

```
TO WHEEL2
CIRCLE2
MOVE.TO.CENTER
SPOKES
END

TO MOVE.TO.CENTER
PENUP
RIGHT 90
FORWARD 40
LEFT 90
PENDOWN
END
```

If you want the wheel to be centered on the screen, draw the spokes first, then move the turtle to the left instead of to the right (you'll have to modify the MOVE.TO.CENTER procedure), and draw the circle.

Here is a useful procedure that takes the radius of a circle as input and draws a 36-sided shape with the correct circumference. Variables are used to make the procedure easier to read and understand. LOCAL makes sure that the variables do not stay in the workspace after the procedure is finished. (Make sure that your PI procedure is in the workspace.)

```
TO CIRC :RADIUS
LOCAL "CIRCUM
LOCAL "DISTANCE
MAKE "CIRCUM 2 * PI * :RADIUS
MAKE "DISTANCE :CIRCUM / 36
BACK :DISTANCE / 2
REPEAT 36 [FORWARD :DISTANCE RIGHT
   10]
FORWARD :DISTANCE / 2
END
```

You can shorten this procedure, avoid over-use of MAKE statements, and produce a more elegant program by writing

```
TO CIRC2 :RADIUS
BACK (DISTANCE :RADIUS) / 2
REPEAT 36 [FORWARD :DISTANCE :RADIUS
   RIGHT 10]
FORWARD (DISTANCE :RADIUS) / 2
END
```

```
TO DIST :R
OUTPUT 2 * PI * :R / 36
END
```

One last wheel program. These procedures draw both the spokes and the rim of the wheel in one operation. I'll leave this for you to explore on your own.

```
TO WHEEL3
REPEAT 20 [SPOKE ARC]
END

TO SPOKE
FORWARD 40
BACK 40
RIGHT 1
END

TO ARC
REPEAT 17 [PENUP FORWARD 40 PENDOWN
   BACK 1 PENUP BACK 39 RIGHT 1
   PENDOWN]
END
```

What next? Think of how you can use what you have learned. You could:

- Put a rind around your wheel and turn it into a grapefruit
- Make a wheel with more (or fewer) spokes
- Take a look at your own bicycle and draw a complete Logo bike
- Make a larger or smaller wheel
- Make a wheel with one spoke that moves around like the sweep second hand of a watch (use PENERASE and WAIT to perfect it)
- Make an 8-spoke wheel and decorate it with pepperoni and mushrooms!

Happy Logo adventures!

A former education and computer consultant, Dorothy Fitch has been the Director of Product Development at Terrapin since 1987. She can be reached at:

Terrapin Software, Inc.
400 Riverside Street
Portland, ME 04103
(207) 878-8200
CompuServe address: 71760,366

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. 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.

NOT JUST ANOTHER CLASS

Dr. Sharon Yoder, editor of the *Logo Exchange* journal, 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 $599.00. For a one-time fee of $1295.00, the package may be obtained with both tape and print duplicating rights, enabling districts to build libraries at multiple sites.

Group Enrollment. A tuition of $206 per participant 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 for $306. Tuition including tape rental is $336. A materials fee of $31 per enrollee is charged for texts and a packet of articles. This fee is waived for enrollees who already have the texts.

Tuition Information, Detailed Course Outlines, and Order Blanks can be obtained from:

LONG DISTANCE LEARNING/ISTE
1787 Agate St., Eugene, OR 97403-1923
Phone 503/346-4414.

Lego, the Curriculum and Children

by Judith Menken

Why LEGO? Why Logo?

Sandy Dawson threw out a startling question at the end of the November 1988 MathWorlds column: "How does LEGO/Logo fit into the curriculum?" (*Logo Exchange*, Vol 7, No. 3) This struck me as close-to-blasphemous appearing in a publication dedicated to "Logo-izing" education.

That question of appropriate application is the litmus test every aspect of my classroom life must pass. Mine is a small alternative public school in East Harlem, New York City. The three primary grades are connected through a theory of child development that allows for unevenness of maturation across different subject areas, as well as a flexibility of structure to accommodate children's individual styles and rates of learning and growth. Every other September, I start over with another group of 6-years-olds; we stay as a group for the next two years. I had this group of 7- and 8-year-olds as well. I had the same group the year before. This luxury of time allows for a gradual organization around all curriculum areas. Given the above, I guess it's clear where Logo fits in, but I have one school computer. Logo is the only software I use.

While the administration has been supportive in terms of supplying the goodies I read about in *LX*, I am on my own as to training, application and assessment.

So, how does LEGO/Logo fit in? For that matter, how does Logo fit in? or just LEGO? or writing process? or the sand table? In fact, what is it we mean when we say "the curriculum"?

I have a moral obligation to teach the children appropriate levels of reading and math. Occasionally, teaching guides filter down, vaguely implying that science and social studies get accommodated as well. And I have my personal interests in art and aesthetic development I want to squeeze in. It takes a bit of juggling to teach the skills the children are expected to acquire along with what I believe is important for them to learn as well as responding to the personal limits and demands of each child's development. The class is run on an integrated curriculum design whereby projects and activities are arranged so that children, sometimes individually, but more often in teams or in small groups, work on several curriculum themes at once. Our lone computer has served as a pivotal agent for many of these projects.

Over the past eight years that I have been battling with learning Logo, I have found it weaving its tentacles through the classroom. It's a natural for the early childhood setting: Pattern block work gets duplicated on screen, geoboard plans are left for others to reproduce more abstractly. The computer also involves not to mention following directions, keyboarding for letter recognition, eye-hand coordination, cooperative learning—all informal and voluntary. Logo, like using the sand table or like reading a book, sometimes provides yet another way to absorb the skills that the school expects.

Now enter into this backdrop LEGO/Logo. I had bunches of fun getting cars to go and lights to blink last summer, but why did my kids need this? And, more difficult yet, is *how*, not to mention *when*, would I turn what is meant for fourth grade and above into "a natural" for the youngsters.

Fitting LEGO/Logo into the curriculum gives it too narrow a scope, just as Logo cannot just be "fit in." The Logo language itself is a metaphor for the larger philosophy of engaging the learner in an active, productive, thinking, planning, self-motivated process. Given that philosophy, I have no problem making certain the childrens' "work period" includes the development of a group "program" for maneuvering the turtle block around the acetate and oaktag "screen" on the rug. In this way, Logo doesn't fit into the curriculum; it is the curriculum.

The Machine Unit

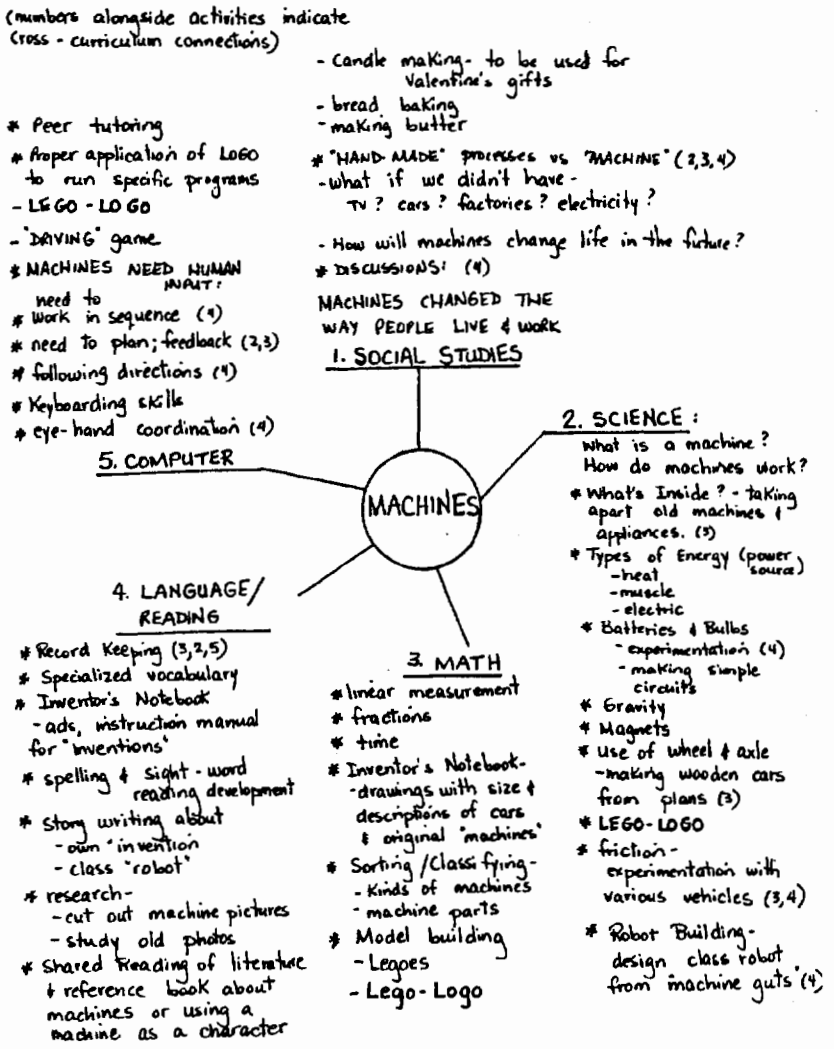
So now the task was how to make LEGO/Logo part of the larger scheme, to give it an integrated place amid the puzzles, woodworking, and upcoming city-wide reading tests.

Basically, due to my limited grasp of the necessary physics, we couldn't focus on constructing mechanically sound units. So I took the perspective of connecting the existing interest, and considerable skill, that kids have with LEGO to a computer based experience. Again, this is not really different than transferring a geoboard design to lines and dots on a monitor, gradually leading children from the concrete to the abstract. Rather than being the end goal of the lesson, the computer serves as one of the tools of that part of the curriculum.

From these ruminations came our unit on Machines. The plans for this unit covered these subject areas through the accompanying activities.

There are obvious cross-overs between many categories. This was a long way to go to merely mention LEGO/Logo. But it could well serve to set the foundation for more directed instruction and exploration in older grades.

The Machines Unit turned out to be a great success on many levels. I'll touch on some of the stronger threads that carried us from January through June. Children have had a lot of experience in building with LEGO. Remember, this is the middle of the second year I have had this group, so everyone has had oodles of chances to freely explore the construction possibilities. I started the unit using their natural interest and and acquired skills. Invention Teams of two or three were assigned to build models of machines with wheels, and then a machine without wheels. Different members of a team then had to build a car and do some basic testing to compare designs. Records were kept, including a drawing of the model and a description of what was tried. For example, a group would try a two-block high ramp and both cars would go down it fine. By the time the ramp was four blocks high William's wheels would keep coming off, but Johnette's didn't. Why? Children began using phrases like "testing" the car, keeping a log of results, making predictions. The written record was being kept in invented spelling with a child in the group chosen as the scribe (writing process).



Introducing the LEGO/Logo car now was a normal next step. Again, children worked in teams. Different tasks might be designing a road and ramp that the car could maneuver over safely, using the basic commands they already had been exposed to in order to travel from point A to point B, or telling the car to GO (number), getting a chance to count the accompanying seconds. Kids spent enormous amounts of time practicing matching their count to the cars' run. The fascination with time (specifically, how long is a second) was a wonderful tangent. I could never have planned for it, and the use of a computer was sort of irrelevant. But in a truly integrated curriculum environment, these kinds of bonuses always pop up.

Initially, the children in my class began to use the DRIVING game after modeling the program with flat blocks and a toy car. Children lay out a track on the rug and take turns giving simple commands. I generally act as the computer, responding to their directions with, "I need more information," or "I don't know how to _____," as needed. This is the format we use whenever introducing a new game I have made or highlighting a new or expanded command.

"Driving" incorporates the previous single-key commands children are now familiar with (F, R, L), while giving the user control of setting up his/her own game. The child designs his/her own road and then "drives" the car along that road with single-key commands, either successfully guiding it to the garage at the end of the road or running off the road and crashing.

This is the first program I had shown the kids using other turtle shapes. In addition to representing the manipulations we were doing with machinery, specifically cars, I hoped to dangle SETSHape in front of some inquisitive souls. Another unspoken goal was to present REPEAT as needed to facilitate the creation of longer roads. The introduction of DRIVING was simultaneous with children using the computer and the LEGO car as well as becoming more adept at their manual LEGO constructions. In addition, other children were making plans on graph paper for original vehicles and then building

them out of milk cartons, with cardboard wheels and straws for axles. As the culminating project for the unit, all children were again on mixed-ability teams that designed and built original inventions, complete with a written manual and ads. All these activities were presented at our Invention Exhibit for parents the last week of school. Children were getting many different experiences from different perspectives using different materials all around the same general curriculum goals. The success and efforts of each experience went towards the quality of experience for others.

As another aside, I had also gotten a LEGO Technics set. As usual, the manual said it was for older elementary school kids. I then spent hours at night painstakingly recreating each of the models so I might be better able to assist children should they want to try it. Again, I misjudged the perseverance and ingenuity of young children when they feel unthreatened and given the opportunity to support each other. The picture of watching Ricky deep in concentration for well over an hour as he made page after page of complex models will always be with me. Knowing that Ricky had terrible attention problems, was a beginning reader who could not focus on an assignment at all, and had a host of birth and home conditions that made success in school difficult at best, just did not fit with the sight before me. The schools must diversify what is considered "curriculum" if these children –and there are more like Ricky than not in our city's schools– are to find real achievement and self-esteem in our system.

In this complex curriculum organization there is no way for the classroom teacher to assess the impact on the class of the computer or its applications. But I do know the kids are turned on to it, they clearly get pleasure from their successes, and they are challenged by their temporary failures. In the case of this Machines Unit it allows the children a broader range of experiences. They wonder, "Why?" and "What if?" Some kids who hadn't yet tapped into the computer are grabbed by the LEGO hook. It makes their toys credible in a grown-up world. LEGO/Logo opens up possibilities and gives kids power to learn in all subject areas through their existing skills and interests. That's what I thought following a curriculum was all about.

Lego car program:

```
to go :seconds
talkto [a b]
onfor :seconds * 10
end
```

```
to turn.l
talkto [a]
setodd
talkto [b]
seteven
talkto [a b]
onfor 40
end
```

```
to turn.r
talkto [a]
seteven
talkto [b]
setodd
talkto [a b]
onfor 45
end
```

DRIVING program:

```
to brush
pd
stamp
forward 3
end

to startup
setup
print [Type ROAD to plan your road.]
print []
print [Type DRIVE when you are ready
   to move your car along your road.]
end

to setup
ct
pu
rg
cc
ht
setbg 1
setsh 1
setc 0
end


to road
ct
pu
cg
setpos [-130 5]
pd
stamp
make "direct readlistcc
run :direct
end
```

```
to done
setc 5
setsh 20
stamp
cc
type [Type DRIVE to move the car
   along the road into the garage.
   Good Luck!]
repeat 8 [type char 32]
end
```

```
to f
brush
end
```

```
to r
right 90
end
```

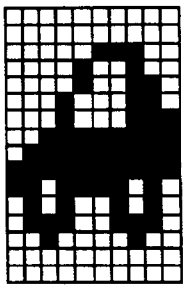
```
to l
left 90
end
```

```
to drive
start.drive
if :direct = "f [seth 0]
if :direct = "r [seth 90]
if :direct = "l [seth 270]
cruise
end
```

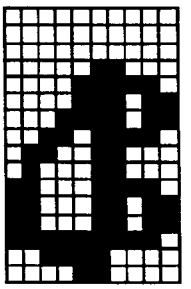
```
to start.drive
cc
pu
setpos [-130 5]
setc 4
setsh 27
st
end
```

```
to cruise
if colorunder = 1 [wait 10 crash
   stopall]
if colorunder = 5 [wait 10 winner
   stopall]
name readchar "key
if :key = "f [forward 3 type "f]
if :key = "b [back 3 type "b]
if :key = "r [right 90 type "r]
if :key = "l [left 90 type "l]
cruise
end
```

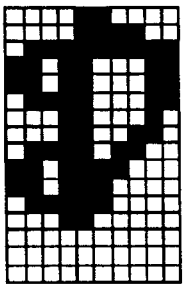
```
to direction
if heading = 90 [setsh 27]
if heading = 0 [setsh 3]
if heading = 180 [setsh 4]
if heading = 270 [setsh 2]
end
```



shape 2



shape 3



shape 4

```
to winner
cg
ht
setpos [-35 0]
label [CONGRATULATIONS!]
setpos [-65 -25]
label [You made it home safely!]
wait 50
startup
end
```

```
to crash
cg
ht
repeat 3 [flash]
setpos [-35 0]
label [YOU CRASHED!]
wait 50
startup
end
```

```
to flash
setbg 4
wait 1
setbg 1
wait 1
end
```

Judith Menken teaches at the East Harlem Block School. She can be reached at

1615 Madison Avenue
NewYork, NY
10029
(212) 567-7744

Logo LinX

From A Different Angle
by Judi Harris

I'll never forget that scene.

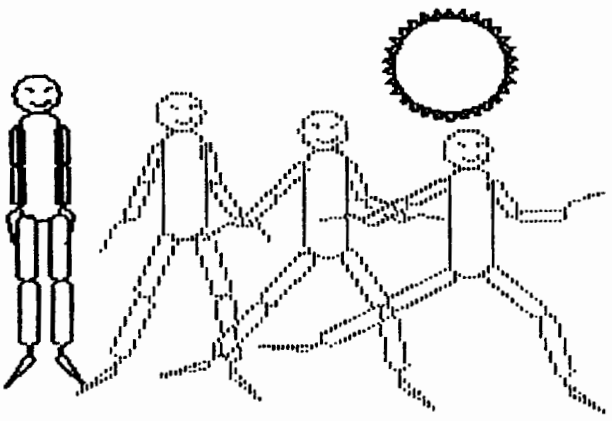
Voice over:
When you've used mathematical principles as a key to enjoyable physical activities, your feeling for mathematics is likely to be warmer, more personal, more engaged.
—Seymour Papert in *Talking Turtle*, a NOVA program produced by WGBH Public Television

Two young girls in black leotards performed a dance to a pulsating rock-and-roll song, mirroring each other's graceful movements.

Voices over:
We wrote the dance like we wrote the procedures for the computer.
—Mandy and Michelle in *Talking Turtle*

As Mandy and Michelle cartwheeled across the floor on my television screen, I felt another set of Logo project ideas begin to percolate in my subconscious.

I dream of helping more children experience mathematics as I do, with all the intimacy of dancing.
—Seymour Papert in *Talking Turtle*



Logo drawing by Larry Robertshaw, Beaver College, 1984

Winter break is over. Many of you are now settling in to several months of uninterrupted school days. The weather is cold enough in many states to move gym classes inside, and physical education teachers can now spend some weeks on gymnastics, dancing, or yoga units. Why not integrate some Logo explorations into these naturally appealing physical activities?

We are all familiar with the facilitative effects of kinesthetic experiences of abstract concepts for learners. Encouraging children to "play turtle" as an integral part of the Logo graphics problem solving process is probably a well-established part of your "facilitator's repertoire" by now. Why not capitalize upon this symbiotic relationship between cognitive and physical activity?

Angular Notions

At one elementary school, it all began with a bulletin board. During one of my "I-need-to-relax" trips to a local bookstore, I happened upon a paperback authored by Olympic gymnast Kurt Thomas (*Kurt Thomas on Gymnastics*, Simon and Schuster, 1980). It contained many beautiful black-and-white photographs of his work on the rings, horse, and floor mats. The precision of his moves and his perfect muscular control were masterfully revealed through these still portraits. The angles formed by his limbs and torso were fascinating.

Books about ballet, yoga, and running seemed to jump out from the shelves. Why not help the children to see the beauty of disciplined bodily movements through geometric angles? I tucked the Thomas book into the crook of my arm, and headed for the cashier.

Viewing Angle

The bulletin board drew many thoughtful observers. I had selected 15 of the full-page pictures of Mr. Thomas that most vividly displayed different body angles, then added semicircles of white self-adhesive label material to identify the angles whose size I wanted the onlookers to estimate. I laminated the pictures, knowing that my students would be wise enough to use their hands to help them to judge angle size. I also hung up a protractor, with directions for its use in measuring angle size. The "answers" were provided under a flap of paper next to each picture.

It was exciting to see students and teachers stop in front of the display and become immersed in the presentation, often for long periods of time, and with no knowledge that they were being watched. The gym teacher asked for similar pictures to hang in the gymnasium, and I was happy to provide her with photographs of children in ballet positions. Students voluntarily began collecting and submitting action shots of football stars, wrestling matches, jazz dancers, and acrobats. Some even brought in cameras to photograph each other on the balance beam and uneven parallel bars.

New Angles

Truly, they had discovered a new way to look at their world. Angle estimation conversations around the displays would often become somewhat heated, and, to the teachers' delight, would be settled with the use of a protractor. It was interesting to watch how this difficult skill was passed on voluntarily from student to student. Many reacted with surprise when they realized that it was not as difficult as it had seemed when they had had to do it in their mathematics books.

It was no surprise, then, when the children's Logo work began to reflect this new area of interest. Some had just discovered the animation tools on the Terrapin Logo utilities diskette (SHAPE.EDIT). Others had found out how to use pen color 0 or penerase (PE) to erase simple shapes just drawn to achieve another type of animation. Still others began exploring text screen animation. Had they been using current versions of Logo, they could also have achieved animated effects with turtle shape changes.

One group of students created pictures of stick figures in all of the basic ballet positions. Another decided to draw ice skating moves. A third group, fascinated with sign language, began a graphic collection of simple signs, such as "I love you."

Personal Angles

I must confess to having been bitten by the same bug. Two projects occupied too much of my out-of-school time during those winter months. One was a "generic gymnast," whose body parts were subprocedures with variable inputs that represented the different joint angles needed to position her in any dance, yoga, gymnastic, or ice skating pose. The other was a traditional four-couple square dance in Sprite Logo from a bird's eye view.

The most energizing aspect of these varied investigations was their common denominator, a quite powerful idea. Seeing the angles of body position and feeling the geometry of movement added depth to the way all of us, students and teachers alike, perceived the world around us. There is a wonder and a balance to the physical experience of abstract notions; a bridge to deeper understanding, no matter what age the thinker.

> *Little by little, we are learning how the computer can make bridges between the mathematical and the sensual; the abstract and the intimate.*
> —Seymour Papert in *Talking Turtle*

(An earlier version of this article appeared in the January 1987 issue of *Logo Exchange*.)

Judi Harris works in the Department of Teacher Education at the University of Nebraska at Omaha as an assistant professor of educational technology. Her teaching, research and service interests include Logo (of course), developmental sequencing in interactive hypermedia materials design, computer-mediated educational telecommunications, and the expressive qualities of children's computer-assisted artwork.

BitNet: JHarris@unomal
Internet: JHarris@Zeus.unomaha.edu
CompuServe: 75116, 1207

MathWorlds

edited by A. J. (Sandy) Dawson

For the past few years David Bell and I have been exploring the use of *LEGO TC logo*. We have used the materials with children as young as five years and as old as late teens—or, if you count myself and David, then children well into their forties, and beyond! This work has been fascinating and yet frustrating. It is fascinating because of the incredible work done by some teachers and some children, and frustrating because we can't seem to put our finger on the critical reasons why this only happens with *some* children and *some* teachers. We can say that the teacher is crucial, but we are still hard pressed to say what exactly it is about the teacher that is crucial. But all this is getting ahead of my story, and that story is about a more formal study we completed using children in grade 5, and the exploratory study that preceded it. That is what I wish to write about this month. Perhaps if you have been working with *LEGO TC logo* you can write me about the responses your children have had as compared to the teachers and children with whom David and I worked.

LEGO TC logo: A study of children's learning

by A. J. (Sandy) Dawson and David Bell

What is this thing called *LEGO TC logo*? Let me try to explain. LEGO has as its root meaning to collect, to pickup, to put together, and later to speak or say. The LEGO in *LEGO TC logo* is the familiar building block toys used so popular with children, with a few important additions. The LEGO materials used in this project were the Technic materials, which include a special computer interface. With this interface, the mechanized objects the children construct, the worlds they literally build, can be connected to the computer by an umbilical cord. Using Logo, children can then program their motorized constructions.

Now, "logos" is the verbal noun of LEGO as well as being the Greek word behind "reason" and "logic."

Hence, if we define reason *broadly*, we could say that humans are rational because they are able to put the world together (LEGO) in a certain way, a way that makes sense (logos) to them. An individual does not have to be able to do a mathematical proof or construct a syllogism in order to be rational in the broad sense.

In *LEGO TC logo*, then, we have an environment that attempts to foster the development of broad reason by providing situations in which children literally *construct* their worlds, in which they turn the images created in their minds into concrete objects—LEGO constructions—which they can animate, mechanize, and program with Logo.

What we wished to find out in our research project using *LEGO TC logo* materials was just what sort of broad reasoning was developed, and whether or not the conjectures made by Ocho, Resnick, and others could be confirmed or refuted.

These conjectures were that:

1. *LEGO TC logo* provides an environment in which students want to experiment and explore.

2. Children learn the method of design and invention. Children readily keep Inventors' Notebooks and make patent drawings. They see themselves as inventors.

3. Students find it easier to learn mathematics and science concepts, and even engineering concepts.

4. *LEGO TC logo* environments give rise to *unexpected* experts. This is especially true for children with strong spatial skills.

5. Children view *LEGO TC logo* as a type of *play*, yet see it as a form of authentic *work*.

6. *LEGO TC logo* is an environment that fosters sharing and cooperation.

7. *LEGO TC logo* is as appealing to males as to females.

Specifically, we wished to address the following researchable questions:

A. Do children using *LEGO TC logo*
   - exhibit science-like behavior?
   - exhibit inventor-like behavior?
   - view themselves as inventors and scientists?
   - learn mathematics and science concepts easily?

B. Do children *play* in their *work* with *LEGO TC logo*?

C. Does the use of *LEGO TC logo* improve the social interactions within a class?

D. Do females get as involved as males in using *LEGO TC logo*?

The Exploratory Study

The exploratory study took place in the fall of 1988 at an elementary school in a semi-rural area just outside Vancouver, Canada. The school had a population of approximately 400 children.

Description of subjects and classes

There were three classes involved in the exploratory study: a grade 4-5 split, a grade 5 and a 6-7 split. A one-day training session introducing *LEGO TC logo* was given to all of the teachers in the study.

The 4-5 split class

The 4-5 split class was under the direction of two teachers. The teacher with the major responsibility for the class (main teacher) was also the computer teacher for the school. He was very experienced with Logo and had an extensive background in its use.

This class contained 13 boys and 15 girls. None of the children had prior experience with *LEGO TC logo*. They were divided into groups of three to four students. In an attempt to balance male and female students who would work well together, these groups were preassigned by the main teacher. After a short time, however, the groups became less stable and routinely students would move from group to group.

The students were involved in the project for approximately six weeks. After an initial session in which the whole class participated (an introductory soapbox derby day), each group had access to the materials for approximately 1.5 hours per week divided over two to three sessions. The students worked with the materials one group at a time in an area of the classroom separated from the rest of the class. During each group's time with the materials, the remainder of the class did regular school work.

The classroom was equipped with one Apple IIe computer (with monochrome monitor and a single disk drive), two Technic 0 kits and two extra pails of bricks. During the soapbox derby day, the class had six Technic 0 kits for their use.

The grade 5 class

The grade five class was the only class taught by a single teacher. She had no prior experience with Logo or *LEGO TC logo*. This class contained 19 boys and 12 girls. None of the children had prior experience with *LEGO TC logo*. They were divided into groups of approximately four to five students. The groups were selected by the teacher in order to achieve a balance between males and females who would work well together. The students remained with the same group throughout.

This class was involved for approximately nine weeks. The first lesson was similar to the grade 4-5 split when the whole class worked on a soapbox derby project. After this, an area was set up in the classroom for *LEGO TC logo* that contained an Apple IIe computer, two Technic 0 kits, and two extra pails of bricks. The students worked in the center, one group at a time on an irregular schedule. Each group had approximately 1.5 hours per week to work with the materials.

During the final week and a half of the exploratory study, this class was given all of the materials (six Technic 0 kits, six pails of bricks and four computers) for an intensive session. The students worked with the materials for five days over the week and a half. The final day concluded with a show-and-tell display for their parents.

The 6-7 split class

The 6-7 split class was taught by two teachers. One (main teacher) was assigned responsibility for the class for three days per week. The other (second teacher) taught the remaining two days of the week. After four weeks of the study, the main teacher went on maternity leave and the second teacher took over full responsibility for the class. Neither teacher had previous experience with Logo or *LEGO TC logo*. The main teacher attended the introductory workshop, and the initial work with *LEGO TC logo* in her class was during her three days per week as teacher. The second teacher was given a half day workshop just prior to the main teacher leaving. The main teacher also attended this workshop and worked with the second teacher.

There were 27 students in this class, 16 boys and 11 girls. They were divided in groups of four to five. The groups were selected by the main teacher in order to achieve a balance between males and females who would work well together. The students remained with the same group throughout.

Students worked with the materials for approximately seven weeks. Like the other classes, the first session involved the whole class, using six Technic 0 kits, on a soapbox derby project. After this an area was set up at the back of the classroom where one group at a time could work with the materials. Each group had access to the materials approximately 1.5 hours each week. The center was equipped with an Apple IIe computer, two Technic 0 kits and two extra pails of bricks.

The Data Sources

Data for the exploratory study came from semi-structured interviews with teachers and students. The interview questions focussed on the research questions identified above.

The grade 4-5 split class

The main teacher was interviewed at the beginning and the end of the study. Interviews were conducted with a random selection of students as the study began. A different random selection of children was interviewed at the end of the study.

The teacher observed that:

- there was some science-like behavior from a few of the children.
- that these children would have exhibited this behavior, with or without *LEGO TC logo*.
- the students tended to copy from others rather than "invent" on their own.
- the students did not act as inventors or scientists, and the students agreed with this observation.

The students however, did express feeling that they enjoyed *LEGO TC logo* more than their traditional science work. The also saw a connection between working with *LEGO TC logo* and science. The teacher did not observe this.

It was generally observed (by both teacher and students) that there was a difference between what the boys and girls did with *LEGO TC logo*. The boys built things like machines with wheels while the girls built houses with lights.

The teacher observed that there was a mix in the effect *LEGO TC logo* had on group interactions. Some groups got along quite well, though others broke up. The students were also mixed on their opinions about working in groups. Some enjoyed it while others didn't.

It was observed by the research team that this class tended to perform poorly in group situations. This did not improve appreciably over the length of the study.

The Grade 5 class

The teacher was interviewed at the beginning and at the end of the study. One group of students was randomly selected to be interviewed early in the study. The same group was interviewed at the end of the study.

The teacher observed that:

- there was an increase in the science-like behavior of the students. The students were much more willing to engage in problem solving, testing, and improvising activities.

- there was an increase in the inventor-like behavior of the students. In her words, the students showed "lots of imagination and were willing to build until they figured it out."

Moreover, the students in this class saw themselves as acting like inventors and scientists when engaged in *LEGO TC logo* activities.

During their interview, the students demonstrated a number of examples that showed an increased understanding of mathematics and science principles that could be traced to their work with *LEGO TC logo*. The teacher was less certain of this and felt that there was a need to more closely define the *fit* of *LEGO TC logo* to the curriculum.

The teacher observed that the girls were more hesitant than the boys at the beginning of the study but this difference largely disappeared over time. However, there remained a difference in the types of things the boys and girls made with *LEGO TC logo*. The girls did the "concession stands" while the boys did the "high decision stuff." One girl observed that "I am not useful with my hands, but I am more useful after *LEGO TC logo*."

The teacher had been working on improving group interactions even before *LEGO TC logo* was introduced into the classroom. She observed that *LEGO TC logo* seemed to assist in this improvement. One student observed that "since I started working with *LEGO TC logo* I have been able to get along with people a lot better."

The 6-7 split class

The main teacher was interviewed as the study began. The teacher who replaced the main teacher was interviewed at the end of the study. One group of students was randomly selected to be interviewed early in the study. The same group was interviewed at the end of the study.

Both teachers observed that:

- the students exhibited science-like behavior when working with *LEGO TC logo*. Students engaged in activities like projecting, experimenting, and learning from the experiments.

- Inventor-like behavior was not evident in this classroom. The students tended to copy from the work of others rather than "invent" new things themselves.

One of the teachers felt that:

- the students saw themselves as creators but not scientists. The children did not have an opinion on this topic.
- the *LEGO TC logo* experience helped the children with the development of math and science concepts.

Both of the teachers and the students observed that there was a difference in the involvement of boys and girls in *LEGO TC logo* use. The girls were less sure of themselves and more reluctant to take chances. One girl, when given the option of continuing working with *LEGO TC logo* or opting out, chose to opt out.

LEGO TC logo was generally seen as having a positive effect on the social interactions in this classroom. One teacher observed that "students did lots of group work which wouldn't have happened otherwise." A student observed that "we learned to work together better."

We made the following observations based on our experience with this exploratory study:

The grade fours did not seem to ever get involved in the work. The decision was made to drop this grade level from the main study.

More computers and more LEGO bricks were needed. The decision was made to give all the available equipment to one class at a time. This also allowed each working group to build projects and to not have to destroy them at the end of the working session.

Working groups needed to be defined and presented with engaging tasks in the spirit of Quasi-Piagetian learning noted by Leron.

We decided to teach the use of *LogoWriter* so that the computer could be used as the tool for keeping student notebooks.

Main Study

The main study took place in the same school during the first few months of 1989. Because of the outcomes of the exploratory study, it was decided to use the grade five teacher from the exploratory study and her new class. Some of the children in this new class would be coming from the grade 4/5 split class used in the exploratory study. Other children would be enrolling in this class from other grade four classes and from out of the school. The composition of the class was as follows: M=male, F=female, E=experienced, N=no experience:

ME 5
FE 7
MN 10
FN 7

The teacher and the researchers discussed how the various working groups should be selected and what composition each group should have. It was decided the the children should be placed in five groups in such a way that as far as possible gender and experience would be balanced. The final composition of the groups was as follows:

A. 2FE, 2MN, 1FN, and 1ME
B. 2FE, 2MN, 1FN, and 1ME
C. 1FE, 2MN, 2FN, and 1 ME
D. 1FE, 2MN, 2FN, and 1 ME
E. 1FE, 2MN, 1FN, and 1ME

Each station had an Apple IIe computer. Located in a central location were eight Technic O kits, six pails of LEGO bricks, assorted extra pieces of LEGO(especially wheels plates) and fifteen motors. The class worked with the materials for six weeks for a total of 31.5 hours per group.

A variety of data collection techniques were used. All the children were interviewed before and after the study began. The teacher was interviewed after the study was over. Students kept notebooks using *LogoWriter*, and these were printed and collected. The teacher developed a questionnaire for her students and gave the research team a copy of the student responses. The teacher also developed a checklist that was included with the student report cards sent home to parents.

Results

It is interesting to note how the children differed in their view of things as compared to the teacher.

For example, the teacher did not think that any *experts*, let alone unexpected experts existed in any of the groups, yet the children in three of the five groups felt that their group did contain an expert. The teacher felt that having experienced and inexperienced students in the groups didn't make a difference. The children agreed, but they added the comment that it was fun to help (to teach?) their inexperienced col-

leagues how to use the materials. And though the teacher concluded that the groups were male dominated and that LEGO was a *male* toy, all that the children would say was that the girls didn't seem particularly comfortable building things with the motors.

When asked what she thought the children had learned, the teacher said she felt that they learned to work cooperatively in groups, that their planning and problem solving skills improved, and they had learned how to use the computer. The children, on the other hand, felt that though they had learned to cooperate more they didn't feel their work in groups had improved all that much. The children did feel they planned better now, but they had no comment about whether or not they felt they were better problem solvers. They did agree they knew much more about using the computer. It is interesting to conjecture as to why the children felt their planning and cooperation skills improved, but that their ability to work in groups did not, particularly since all the planning and cooperation took place in groups!

Both the teacher and the children felt that groups of six children was too large.

Regarding the work/play issue, the teacher said that the *LEGO TC logo* sessions were play that at times was difficult, implying that when it became difficult it was closer to being work. She did say that she thought it was productive play, and not frivolous play. Five of the six groups said the *LEGO TC logo* was fun, and at times seemed like work and at times like play!

In comparing her experience over the two years, the teacher felt that the second time around she provided more structure for the children, gave them more challenges and tasks, and generally pushed them more. The children with experience from the previous year said that their experience the second time around was harder, more challenging, but more interesting, and better organized. They said that the existence of more equipment was a vast improvement over the previous year.

As this report goes to press, we will be in the midst of another study of *LEGO TC logo*, but this time the focus is on how to integrate *LEGO TC logo* into the teaching of language arts, science, and social studies so that the *LEGO TC logo* system becomes part of the materials and approaches to the teaching and learning of these subjects, rather than being the object of study itself.

References

Ackermann, E. (1987). *LEGO logo activities: A formative evaluation.* Boston: MIT Media Laboratory.

Ocko, S., Resnick, M., & Ackermann, E. (1986). *LEGO TC logo*. Proceedings of Logo '86. Boston: MIT Press.

Ocko, S., & Resnick, M. (1987). *Integrating LEGO with logo: Making connections with computers and children.* Boston: MIT Media Laboratory.

Resnick, M., & Ocko, S. (1987). *LEGO TC logo and science education.* Boston: MIT Media Laboratory.

Resnick, M. (1988). MultiLogo: *A study of children and concurrent programming*. Unpublished Master's Thesis, Boston:MIT, Department of Electrical Engineering and Computer Science.

A. J. (Sandy) Dawson is Director of the Professional Development Program and an Associate Professor in the Faculty of Education at Simon Fraser University. David Bell is Director of the Center of Educational Technology in the Faculty of Education at Simon Fraser University. Their email addresses are userdaws@sfu.bitnet and userbell@sfu.bitnet respectively.

Logo Connections

Logo Robotics

by Glen L. Bull and Gina L. Bull

In this year's columns, we are discussing practical multimedia suggestions for Logo. In each column, we are reviewing a different multimedia option that can enhance Logo. Thus far, suggestions for companion programs and peripherals have included:

- an upgrade of Logo to the most recent version, so that you can take advantage of the full graphics and text capabilities of your computer
- acquisition of a paint program for use as a "graphics editor" of pictures that can then be imported into Logo
- acquisition of a video digitizer for capture of "digital photographs" that can be used as graphics screens in Logo

These first three suggestions are comparatively inexpensive. Depending on the version of Logo you are using, it may be possible to upgrade to the most recent version for as little as $50.00. A good paint program can be obtained for less than $100.00, and a video digitizer can be acquired for $129.00.

Logo Motors and Sensors

The term "multimedia" refers to the use of several different media in combination. In this column, we are going to discuss Logo robotics. The combination of a computer with external levers, gears, motors, pulleys, and sensors certainly qualifies as a multimedia environment. As many *Logo Exchange* readers know, *LEGO TC logo* is a robotics kit that uses the Logo programming language to control LEGO motors and sensors. There are two versions of this kit. The "official" product (*LEGO TC logo*) distributed by the LEGO corporation consists of a modified version of *LogoWriter* and an accompanying computer card and interface that allow motors and sensors to be connected to the computer. A variation of this concept consists of an interface developed by Terrapin that connects to the Apple game port, and can be used to control LEGO motors and sensors. The Terrapin product controls fewer motors and sensors at one time than the official LEGO kit, but is also less expensive.

LEGO TC logo has a distinguished heritage. The names, differing by only one vowel, sound as though they should go together. *LEGO TC logo* was developed in the M.I.T. Media Laboratory. Recently the ties between the LEGO corporation and Logo were strengthened when the Danish headquarters of the LEGO corporation presented Seymour Papert with an endowed chair at M.I.T.

There are references to use of Logo to control and interact with the external environment in many of Papert's early writings. In one suggested experiment, for example, a Logo program would be used to balance a broom handle placed on a wagon. Sensors would be used to detect when the broom was about to topple, and Logo-controlled motors would control the movement of the wagon to prevent this from happening. In another experiment, Papert suggests development of a Logo program to control the motion of a yo-yo. Seemingly simple experiments with common objects prove surprisingly difficult to implement in practice, providing a rich environment for exploration of a wide range of concepts. To our knowledge, no one has yet used Logo to conduct either of these two experiments first suggested more than two decades ago. However, LEGO-Logo comes closer to fulfilling this early vision of Logo than any other product.

LEGO TC logo is packaged in a fashion similar to *LogoWriter*. Each product is accompanied by a box of instructional materials, projects, and suggested applications. *LogoWriter* has activity cards. *LEGO TC logo* has two sets of materials for each project. A pamphlet shows how the LEGO bricks and components are assembled on a step-by-step basis. An accompanying programming booklet describes Logo programs and activities that can be developed once the LEGO components are assembled.

The assembly instructions are clear and concise. Over a period of years, we have worked with several hundred teachers and students using *LEGO TC logo* for the first time in classes and workshops. Many of them, of course, had previously played with sets of LEGO bricks. Even those who with no prior experience seem to encounter little or no difficulty. After only a short introduction, most individuals are also able to follow the directions for creating Logo programs to control the LEGO devices they have created. We find that it is a good idea to suggest one of the simpler projects, such as the LEGO traffic light, before tackling a more complex project such as the LEGO robot turtle. Often the projects lend themselves to groups working in teams. It is a tribute to the quality of the accompanying instructional materials that almost everyone is ultimately successful.

Logo was developed to provide an educational interface for computers at a time when classroom use of computers was regarded as unusual. The Logo turtle provided an "object to think with." Today, however, almost everyone in an academic environment routinely uses microcomputers. In that context,

adults do not always immediately see the power and educational potential of turtle graphics. In our experience, a class of adult learners left to their own devices may sometimes investigate a few of the *LogoWriter* activity cards and then find other enterprises with which to occupy themselves. Many teachers now have the incorrect perception that Logo is only suitable for elementary grades, and sometimes make unfavorable comparisons with other word processing and graphics tools that they are already using.

When LEGO-Logo is introduced, it is sometimes difficult to get the class to leave at the end of the period. The inherent interest in LEGO-Logo is motivating, and students frequently go beyond the minimal requirements of the assignment. This activity immediately dispels the inaccurate one-dimensional perception of Logo, and can provide a useful bridge to other aspects of Logo. For this reason, LEGO-Logo can be one of the best possible introductions to Logo for adults. Often we will provide an initial class on turtle graphics, and then introduce LEGO-Logo in the second session.

Aside from its intrinsic appeal, the marriage of LEGO and Logo has a great deal to recommend it. The world of computers tends to operate in a binary fashion. Either the syntax of a programming instruction is correct, or it is not. In contrast, the analog world of LEGO parts is characterized by successive degrees of correctness. A LEGO pulley may be in the correct position, but can need a slight nudge so that it turns without binding. In this analog world children can learn the important skills of tinkering, problem solving through successive approximation, and hands-on experimentation.

In the not too distant past, a generation of Americans learned about tinkering through experimentation with vacuum-tube radios and backyard automotive repairs by "shade-tree" mechanics. Today's radios have surface-mount integrated circuits, and pollution equipment on car engines makes a tune-up a job for a trained mechanic with specialized equipment. LEGO-Logo provides a an environment for tinkering. Another appealing aspect of LEGO-Logo is that it appears to be equally appealing to children regardless of gender. Certain disciplines, particularly in engineering, have been dominated by males. Experiences with a LEGO-Logo environment can allow all children to learn the appeal of designing and constructing a machine from start to finish.

LEGO TC logo Enhancements

LEGO TC logo is a modification of the first version of *LogoWriter*. This version of *LogoWriter* was developed for a non-standard operating system designed by Logo Computer Systems, Inc (LCSI) to permit *LogoWriter* to fit into a 64 kilobyte memory space. As a result, *LEGO TC logo* files can not be stored on either a standard DOS 3.3 or ProDOS Apple disk. Instead, they must be stored on a specially formatted *LEGO TC logo* disk. As Apple II computers with 128 kilobytes became more prevalent, Version 2 of *LogoWriter* was upgraded to function under the standard ProDOS operating system. However, *LEGO TC logo* has never been updated.

LEGO TC logo is also missing some of the standard *LogoWriter* commands. Informal discussions with personnel at both LEGO and LCSI suggest two reasons for these omissions. One reason may have been simply to save space in order to make it possible to add the *LEGO TC logo* commands to the *LogoWriter* language. Personnel at LEGO also indicate that LCSI omitted some of the *LogoWriter* commands because of a concern that teachers who purchased *LEGO TC logo* would not purchase *LogoWriter* as well. Whatever the reasons, these omissions highlight an area in which *LEGO TC logo* could be enhanced.

The non-standard disk format makes it cumbersome to import external pictures created with paint programs or captured with digitizers into *LEGO TC logo*. It also means that students must carry two types of disks—one for *LEGO TC logo* programs, and another for regular Apple files. Failure to include all of the *LogoWriter* commands in *LEGO TC logo* means that students cannot easily combine many of their regular *LogoWriter* programs with *LEGO TC logo* activities.

LogoWriter was upgraded to ProDOS years ago. The LEGO corporation should enhance an already excellent product by upgrading it to the same standard. The ability to share the same disk used for storage of other Apple files is a long overdue improvement.

Inclusion of the omitted *LogoWriter* commands is a separate matter. The concern that *LEGO TC logo* could displace *LogoWriter* sales is a legitimate issue. On the other hand, there are possible ways to provide consumers with an uncrippled version of *LEGO TC logo* while protecting legitimate economic concerns. For example, *LEGO TC logo* kits could be packed with a certificate which would allow those possessing a legitimate *LogoWriter* license to redeem it for a version of *LogoWriter* with *LEGO TC logo* extensions.

Unless a solution to the current limitations are identified, users face the prospect of a *mono-media* environment rather than a *multimedia* environment. In this mono-media environment, a separate specialized version of Logo would be required for each medium—one for use with *LEGO TC logo*, a

second for word processing and language applications, a third for control of videodisc players, and other peripherals.

Programs such as *HyperCard* provide one possible model for multimedia extensions. Many different vendors sell extensions to *HyperCard* that enhance its range of capabilities. This permits a single version of *HyperCard* to be used for a variety of hypermedia applications: control of videodisc players, CD-music discs, telecommunications, graphics applications, and so forth. It would greatly benefit users if a similar approach could adopted for Logo. Logo was one of the first widely used *extensible* programming languages. This extensibility permitted user-defined procedures to be used as though they were built-in commands. Provision for addition of commands developed for different multimedia extensions would easily fall within the tradition and heritage of the language.

Innovative *LEGO TC logo* Projects

If two geographically separate sites possess *LEGO TC logo*, it can easily be the focus of a telecomputing project. One project of this kind involved students at Murray High School in the Albemarle School System in Virginia, and students at School 57 in Moscow. Sylvia Weir carried a *LEGO TC logo* kit donated by Tom Lough of the U.S. branch of LEGO to Moscow on a trip to Russia.

Despite concerns about whether customs officials on both sides might look askance at importation of this specialized electronic equipment, the kit was safely delivered. The two classes communicated with one another via the San Francisco-Moscow teleport. Students at one site developed descriptions of construction projects that the other class attempted to replicate. Later the Albemarle County teacher, Becky Fisher, was able to visit her Russian friends.

On a smaller geographic scale, similar projects are under way in other Virginia schools. For example, Mano Talliaver, at the Science and Mathematics Center in Richmond, and Tom Morgan, director of the Central Virginia Governor's school in Lynchburg, are developing plans to permit teachers and classes in their respective areas to interact on joint *LEGO TC logo* projects using Virginia's public school telecomputing network as the vehicle for communication. Coordinating efforts of young engineers at different geographic sites adds an entirely new dimension to such projects. However, it is useful preparation for the globally interconnected world of the 21st century.

There are other innovative ways in which teachers have conducted LEGO-Logo projects. Except at the LEGO factory, there are seldom all the pieces that might be wanted for a project. Some LEGO pieces such as motors and certain types of gears are usually in particularly high demand. One innovative teacher made use of the relative rarity of certain LEGO pieces in a project combining engineering, economics, and social studies. Teams of students were required to bid on different construction projects. Each team was given the same budget; rarer LEGO pieces were priced accordingly. Students were allowed to use electronic spreadsheets to develop their bids. The effect of economics on engineering and construction techniques was then the topic of discussion in a social studies unit. This type of multimedia, multidisciplinary project typifies the use of technology to enhance existing curriculum objectives.

The Third Dimension

Hypertext applications give a third dimension to written materials. LEGO/Logo allows use of a third dimension for experiments that previously took place with a turtle on a two-dimensional monitor. Papert envisioned this type of three-dimensional experimentation over 20 years ago when he conceived the first robotic turtles. Perhaps the developer of LEGO envisioned some means of allowing LEGO machines to be programmed by their young inventors. The alliance of LEGO and Logo carries both beyond their initial boundaries.

Information about *LEGO TC logo* can be obtained from LCSI: 1-800-321-5646 or from LEGO Systems, Inc., 1-800-527-8339.

Information about the Terrapin LEGO kit can be obtained by calling Terrapin at 2-7-878-8200

Glen and Gina Bull
Curry School of Education
Ruffner Hall
University of Virginia
Charlottesville, VA 22903

BITNET addresses:
Glen: GBULL@ VIRGINIA. Gina: GINA@VIRGINIA.

Date, Time, Palette, and Diskspace:

Four New Primitives for LogoWriter on the IBM/PC

by Charles E. Crume

LogoWriter was designed to work in a similar manner on several brands of microcomputers. Primitives that are easy to implement on one machine are sometimes omitted on other machines because of hardware differences. This article presents four useful procedures that enhance the IBM/PC version of *LogoWriter*. The four procedures are: (a) DATE, (b) TIME, (c) PALETTE, and (d) DISKSPACE. These procedures report the current date, report the current time, change the color palette on a Color Graphics Adaptor (CGA), and report the number of free bytes on a disk. They make use of the primitives .DEPOSIT, .EXAMINE, .INTERRUPT, and .REGS, and call the operating system interrupts 33 (21h) and 16 (10h) of MS-DOS, therefore, they will only work on the IBM/PC version of *LogoWriter*.

The first procedure reports the date currently in the computer's memory (the date is normally set during the process of turning on the computer). This procedure, given below, can be used by children to include the date in their program output.

```
TO DATE
.DEPOSIT .REGS 10752
.INTERRUPT 33
OUTPUT (SENTENCE ITEM INT (.EXAMINE
   (.REGS + 6)) / 256 [January Febru-
   ary March April May June July
   August September October November
   December] WORD REMAINDER (.EXAMINE
   (.REGS + 6)) 256 ", .EXAMINE .REGS
   + 4)
END
```

To print the current date, use the command shown below:

```
PRINT DATE
```

the output would be a date such as:

```
August 27, 1989
```

the command:

```
(PRINT [TODAY'S DATE IS:] DATE)
```

would print:

```
TODAY'S DATE IS: August 27, 1989
```

To include the date onto a graphics screen, position the turtle to the desired location, then use the command:

```
LABEL DATE
```

The second procedure, called TIME, reports the time currently in the computer's memory (the time is normally set during the process of turning on the computer). TIME calls the procedure .TIME which does the majority of the processing.

```
TO TIME
.DEPOSIT .REGS 11264
.INTERRUPT 33
OUTPUT .TIME .EXAMINE .REGS + 4
   .EXAMINE .REGS + 4 .EXAMINE .REGS
   + 6
END

TO .TIME :HOURS :MINUTES :SECONDS
MAKE "HOURS INT :HOURS / 256
IF :HOURS < 10 [MAKE "HOURS WORD "0
   :HOURS]
IFELSE :HOURS < 12 [MAKE "HOURS WORD
   :HOURS "am] [MAKE "HOURS WORD
   :HOURS "pm]
MAKE "MINUTES REMAINDER :MINUTES 256
IF :MINUTES < 10 [MAKE "MINUTES WORD
   "0 :MINUTES]
MAKE "SECONDS INT :SECONDS / 256
IF :SECONDS < 10 [MAKE "SECONDS WORD
   "0 :SECONDS]
OUTPUT (WORD BUTLAST BUTLAST :HOURS
   ": :MINUTES ": :SECONDS LAST
   BUTLAST :HOURS LAST :HOURS)
END
```

To print the current time, use the command shown below:

```
PRINT TIME
```

the output would be the time, such as:

```
12:51:33am
```

the command:

```
(PRINT [THE CURRENT TIME IS:] TIME)
```

would print:

```
THE CURRENT TIME IS: 12:51:33am
```

The third procedure changes the color palette on an IBM CGA. The CGA has limited color capability (three foreground plus background) in its 320x200 pixel mode (whereas the EGA has 15 colors plus background, and the VGA even more). When LogoWriter on a CGA starts, it defaults to the color palette of white, magenta, and cyan. There is another palette comprised of the colors orange (sometimes yellow or brown depending on the equipment), red, and green. Unfortunately, LogoWriter does not provide a primitive to switch between the two palettes. To switch between the two palettes the following procedure, called PALETTE, can be used. The procedure requires a single input - the number of the color palette to be used. The value zero is for white/magenta/cyan and the value one is for the orange/red/green palette. The procedure is error trapped to ensure a valid number is supplied.

```
TO PALETTE :X
IF NOT OR :X = 0 :X = 1 [(TYPE [PAL-
   ETTE DOES NOT LIKE] :X [AS INPUT]
   CHAR 13) STOPALL]
.DEPOSIT .REGS 2816
.DEPOSIT .REGS + 2 256 + :X
.INTERRUPT 16
END
```

The final procedure, DISKSPACE, reports the number of free bytes on the specified disk. The procedure requires a single input - the letter of the disk drive to be reported. The procedure is error trapped to ensure a letter between "A" and "Z" has been supplied (upper or lower case is OK). The procedure also error traps non-existent drives and drives that are not ready (door open, no disk inserted, etc.).

```
TO DISKSPACE :DRIVE
IF AND (ASCII :DRIVE) > 96 (ASCII
   :DRIVE) < 123 [MAKE "DRIVE CHAR
   (ASCII :DRIVE) - 32]
IF NOT AND (ASCII :DRIVE) > 64 (ASCII
   :DRIVE) < 90 [(TYPE [DISKSPACE
   DOES NOT LIKE] :DRIVE [AS INPUT]
   CHAR 13) STOPALL]
.DEPOSIT .REGS 13824
.DEPOSIT .REGS + 6 (ASCII :DRIVE) -
   64
.INTERRUPT 33
IF (.EXAMINE .REGS) = 65535 [(TYPE
   [DRIVE] :DRIVE [IS INVALID OR NOT
   READY IN DISKSPACE]) STOPALL]
OUTPUT (.EXAMINE .REGS + 2) * (.EXAM-
   INE .REGS) * (.EXAMINE .REGS + 4)
END
```

To determine how many bytes are available on the A disk drive, use the command:

```
PRINT DISKSPACE "A
```

the result will be a decimal number such as:

```
131072
```

The DISKSPACE procedure is more appropriately used in a conditional statement to determine whether enough space is available for disk operations such as saving text files. An example would be:

```
IFELSE (DISKSPACE "A) > 10000
   [SAVETEXT "MYTEXT] [PRINT [THERE
   IS NOT ENOUGH ROOM ON THE DISK]]
```

If there were more than 10,000 bytes of space available, the above statement would save the contents of the text screen in the file MYTEXT. If there were not 10,000 bytes of disk space, a message indicating such would be displayed.

Charles E. Crume
Factorum Software
P. O. Box 8874
Reno, Nevada 89507

Extra For Experts

edited by Mark Horney

But Can I Shoot Down Aliens with It?

An Assembly Language Enhancement for Logo

by Ron Hackett

As a devout Logo fanatic, I am very familiar with the many powerful (and fun) features of Logo and *LogoWriter*. However, whenever a student asked "But can I shoot down aliens with it?", I had to admit that Logo's TONE command was not quite up to the task. No self-respecting alien could be fazed by a simple BEEP or two! Clearly, what is needed is a PHASOR command. After several frustrating attempts to produce a suitable sound using TONE with variable inputs, I realized that the only acceptable solution would be to write a short assembly language program and to access it with *LogoWriter's* .CALL command.

Assembly language may be a topic that arouses fear (or indifference) in the hearts of many Logo users, but you don't have to figure out how the assembly language portion of the program works in order to effectively defend yourself against the Alien Invaders. Carefully type the following procedures onto a *LogoWriter* page and then type INSTALL.PHASOR in the Command Center. When the cursor returns, try UP, DOWN, and PHASOR for some exciting new Logo sounds! You can also experiment with your own procedures combining various sequences of UP and DOWN to produce new and improved alarms, sirens, and phasors.

PHASOR Program Listing for Apple *LogoWriter* Version 2.0:

```
TO  INSTALL.PHASOR
MAKE "AL [162  0  173  48  192  138  168
   136  208  253  202  208  245  96  162  0
   173  48  192  138  168  136  208  253  232
   208  245  96 ]
INSTALL  39680  :AL
END

TO  INSTALL  :HERE  :CODE
IF  EMPTY?  :CODE  [STOP]
.DEPOSIT  :HERE  FIRST  :CODE
INSTALL  :HERE  +  1  BUTFIRST  :CODE
END

TO UP
.CALL  39680
END

TO  DOWN
.CALL  39694
END

TO  PHASOR
REPEAT  8  [UP  DOWN]
END
```

How the Program Works:

LogoWriter leaves a small section of free memory in Apple II computers beginning at decimal location 39680. The AL list is a decimal translation of two short assembly language routines, one for the UP sound and one for DOWN. INSTALL is a simple recursive procedure that actually .DEPOSITS the code into memory, starting at location 39680. UP and DOWN .CALL the assembly language routines that produce their respective sounds, and PHASOR is just a repeated sequence of the two. The complete assembly code is presented at the end of this article for interested assembly language programmers.

That's all there is to it! INSTALL.PHASOR provides a useful and fun addition to *LogoWriter's* sound capabilities. Of course, you can automate the process by using STARTUP, and you can make your new sounds available on every page on the scrapbook disk by setting them up as TOOLS. Another possibility is to .BSAVE the assembly language code once as a binary file on the scrapbook disk, then .BLOAD it in a STARTUP page. However, it actually takes *LogoWriter* longer to .BLOAD the file than it does to install it directly into memory, so I opted for the INSTALL.PHASOR approach.

PHASOR is only one possibility for sound enhancements to Logo. Small assembly language routines could also provide warbling sirens, gun shots, whistles, and many other interesting sound effects. The only limitations are the imagination of the programmer and the amount of free memory in the Apple computer!

Changes For Other Versions of Logo:

The program will also work with Apple *Logo II*, LCSI *Logo II* and Terrapin Logo. Just make the appropriate modifications as listed below. I don't have access to *Logo PLUS*, but I assume it would work there as well. You would need to know where some free memory space is located and modify the code accordingly. The required modifications would also be relatively simple for Commodore Logo and *LogoWriter*,

because Commodore machines use the same assembly language as the Apple II series. IBM Logo, however, would require entirely different assembly language code. It would be a great project for a Logo user who is also experienced in IBM assembly language. (Is anyone out there?)

Apple Logo II:

1 In INSTALL.PHASOR,
   a add the following line after the MAKE statement:
   .DEPOSIT 768 45
   b retype the last line as INSTALL 23552 :AL
2 In INSTALL, change EMPTY? to EMPTYP
3 In UP, retype the line as .CALL 23552
4 In DOWN, retype the line as .CALL 23566

LCSI Logo II:

1 In INSTALL.PHASOR,
   a add the following line after the MAKE statement:
   .DEPOSIT 0 768 45
   b retype the last line as INSTALL 23552 :AL
2 In INSTALL,
   a change EMPTY? to EMPTYP
   b retype the .DEPOSIT line as .DEPOSIT 0 :HERE FIRST :CODE
3 In UP, retype the line as .CALL 0 23552
4 In DOWN, retype the line as .CALL 0 23566

Terrapin Logo:

1 In INSTALL.PHASOR, retype the last line as INSTALL 39328 :AL
2 In INSTALL, retype the IF statement as IF EMPTY? :CODE THEN STOP
3 In UP, retype the line as .CALL 39328 0
4 In DOWN, retype the line as .CALL 39342 0

Assembly Language Code:

```
9B00 A2 00    UP    LDX  #$00   Swoop up.
9B02 AD 30 C0 LOOP1 LDA  $C030  Toggle the
                                speaker.
9B05 8A             TXA
9B06 A8             TAY
9B07 88       LOOP2 DEY
9B08 D0 FD    BNE   LOOP2       Output one
                                pitch.
9B0A CA             DEX         Raise the
                                pitch.
9B0B D0 F5    BNE   LOOP1       Loop until
                                finished.
9B0D 60             RTS         Return to
                                Logo.
9B0E A2 00    DOWN  LDX  #$00   Swoop down.
9B10 AD 30 C0 LOOP3 LDA  $C030  Toggle the
                                speaker.
9B13 8A             TXA
9B14 A8             TAY
9B15 88       LOOP4 DEY
9B16 D0 FD    BNE   LOOP4       Output one
                                pitch.
9B18 E8             INX         Lower the
                                pitch.
9B19 D0 F5    BNE   LOOP3       Loop
                                until finished.
9B1B 60             RTS         Return to
                                Logo.
```

Expand Your Horizons!

For stout-hearted Logo users who would like to venture into the fascinating and powerful world of Apple assembly language, the following books provide a good introduction:

De Jong, M. (1982). *Apple II assembly language*. Indianapolis, IN: Howard W. Sams & Co., Inc.

Gilder, J. (1985). *Now that you know Apple assembly language:What can you do with it?* Brooklyn, NY: DataCraft, Inc.

Ron Hackett is a Microcomputer Specialist with the Special Education Division of BOCES 2, which serves students and school districts in central Suffolk County, Long Island, New York. In this position, he is extensively involved in staff training and spreading the Logo and LEGO/Logo word. He also teaches graduate courses in Logo programming for educators at the State University of New York at Stony Brook and Dowling College in Oakdale, New York. A confirmed hardware and software hacker at heart, Ron is also co-president of R & D Microsystems, a small independent company specializing in microcomputer peripherals and software for education and fun.

Ron Hackett, Ph.D
105 Whittier Place
Port Jefferson, NY 11777

Logo: Search and Research

Strategies for Learning Logo
by Douglas H. Clements

"Johnny can do anything, but Ralph just can't seem to get the hang of it."

Does that sound familiar? In most classrooms, some students learn Logo programming much better than others. Why?

Certainly, there are many reasons. Most students receive limited amounts of instruction. They have different backgrounds of exposure to computers. Many teachers of programming are new to the enterprise. Further, we are just now learning how to teach this complicated and challenging ability.

Still, such reasons do not go very far in explaining what successful learners *do* that helps them learn. One group of researchers suggested that students bring different *patterns of learning* to the programming context (Perkins, Hancock, Hobbs, Martin, & Simmons, 1986). They observed high-school and intermediate-grade students learning programming. They found that the *way* students managed their programming tasks had a great impact on their success in both solving the problem and learning. The results of their small studies are tentative, but interesting.

A Non-Trivial Pursuit

Before discussing students' learning patterns, the researchers emphasize that the activity of programming is surprisingly difficult. Even seemingly "simple" problems demand that students invent new ways to use the commands of the language. The goal is to teach students to solve programming problems that are more than trivially different from programs they have already seen. So, we are expecting *far* transfer of learning, which is notoriously difficult to achieve.

For example, the researchers asked students to write a program that would draw three identical rectangles in a stack, with some space in between. Beverly wrote the code for one. The researcher asked how she could use it to create the other two. She ran the procedure several times, moving the turtle to place it in different locations. "Can I put this into a procedure?" she asked, referring to the three repetitions she had produced. This was the goal, of course. Beverly, however, had never written embedded procedures, nor had she ever seen one! So, she was being quite creative. The difficulty is that the road to programming is replete with pitfalls. Successful students must both act creatively and avoid the pitfalls. What are some of these pitfalls?

Stoppers and Movers

What do you do when you are programming and it is not clear what to do next? Should you proceed by trial and error, try to break the problem down, or something else? Many students consistently do what is simplest: They quit. They disengage from the task whenever trouble occurs. These are the stoppers. Stoppers appear to abandon any and all hope of solving the problem on their own. For example (Perkins, et al., p.42):

Student:	I'm stuck.
Observer:	What do you think the problem is?
Student:	I don't know how to program it.
Observer:	What ideas do you know?
Student:	I don't know.

Stoppers become frustrated every time they make a mistake or encounter a problem. They don't keep close track of what their programs do as they write them. They often repair buggy programs by making haphazard changes. They have difficulty breaking problems down into parts.

Why does this pattern emerge? Feelings may have a lot to do with it. Novices are most likely to feel unsure of what they are doing. They doubt their control over the activity. So, the activity may be a threat to their self-esteem. Making matters worse, stoppers have a distinct attitude toward making mistakes. They see mistakes as measures of their worth. They do not believe, as some of their peers do, that mistakes are part of programming and of learning (see Papert, 1980). Such strong feelings exacerbate stoppers' unfavorable patterns of learning.

Many stoppers disengage so completely that they learn little. The researchers found, however, that many times stoppers needed only a small amount of encouragement to complete a task. For example, Tom insisted he did not know what he was doing. He frequently quit when faced with any bug. With one simple prompt from an observer, however, Tom proposed an idea about an error message and tried it out successfully. With encouragement and instruction, students might learn not to give up too easily.

What about movers? These students try one idea after another, writing or revising their program and testing it, never stopping long enough to appear stuck. Movers may do well—they are at least trying something—but extreme movers move too fast, without reflection. They go around in circles, often repeatedly trying approaches that they have already found did not work. Such movers are also disengaging from the prob-

lem. Instead of thinking hard about a bug and the reasons behind it, they quickly move on and make yet another change.

The Untapped Power of Close Tracking

A critical process in computer programming is "running the program in your head." Called "hand tracing," or close tracking, this process is useful for ferreting out bugs both before and after running a program. Tracking demands that one takes the perspective of the computer. In addition, it demands knowledge of the language's primitives, flow of control, and how these affect the state of the system (e.g., the position and orientation of the turtle).

In contrast to its importance, students do not use tracking much. According to the researchers, avoidance of tracking is akin to the tendency not to check work on mathematical problems. Most seem not to recognize its usefulness in either situation. The avoidance may also stem from a lack of confidence based on the belief that "you can never be sure about what the computer will do." In addition, it may result from a faulty understanding of how the programming language works. As reported in previous columns, many students hold misconceptions such as, "If a square procedure has FD 90 RT 90, a smaller square would be FD 30 RT 30."

Finally, as in proofreading their compositions, people overlook errors when reading their computer code because they project their expectations onto the code. This is merely another way to say that they do not take the perspective of the computer. (Recall the "superbug" of believing that there is a hidden mind somewhere in the language that has intelligent, interpretive powers; see the Februray 1989 column, "To Err is Human...To Debug, Divine.") For example, one student was asked to close-track a rectangle procedure. He was unable to maintain the correspondence between the code and the features of his drawing. He would start out tracking precisely but, as time when by, he drifted and began assuming that the code was going to do what he expected it to do.

The Dangers of Tinkering

The researchers also observed a lot of "tinkering"—writing some code and then repeatedly making small changes in an attempt to get it to work. For example, Donald was attempting to solve the three rectangles problem. He wrote a line of code using two procedures he had defined previously.

```
REPEAT 3 [REC MOVE REC MOVE REC MOVE]
```

This produced nine rectangles. Instead of thinking about the instruction and how REPEAT works, Donald assumed that his code needed some minor change. He made repeated unsuccessful efforts to fix the bug by rearranging the order of the procedures in the REPEAT list.

Tinkering, used mostly by movers, is potentially useful. It is natural and productive for some children at a certain developmental stage (Noss, 1984; Papert, Watt, diSessa, & Weir, 1979). *Effective* tinkering, however, depends on close tracking, and movers often do not track. Further, even if tinkering achieves the students' goal of producing a drawing, the haphazard approach usually does *not* achieve the greater goals of reflection and learning. Further, the resulting program may be more patchwork than planned (structured).

The Necessity of Breaking Problems Down

Also critical is the ability to break a problem into manageable pieces. Too often, however, children engage in immediate-mode tinkering that is more like video game maneuvering than programming. Children often continue by writing one long procedure from that code, complete with all the false moves, backups, and erasures of the original. This requires mere stamina, not higher-order thinking.

As with tracking, children frequently do not recognize that breaking the problem down will help. Even if they try to use this strategy, they may not be able to recognize suitable chunks. They often use visual rather than analytic strategies (see December/January's column, "Strategies for Solving Turtle Geometry Problems"). Decomposition is yet more difficult because students often lack Logo programming templates (also described in previous columns). Even more difficult is decomposing in the middle of coding, instead of planning in advance. For instance, Rodney began working on the three rectangles problem by writing a RECTANGLE procedure that drew a rectangle and moved the turtle to the top of the screen. Naturally, he found it difficult to use this procedure to make the other two rectangles. His debugging was further plagued by trouble stemming from the arbiter way he had decomposed the problem originally.

Suggestions for Teaching

The researchers offer several teaching suggestions. Some are simple, but make sense. For example, we should discuss and emphasize the value of certain practices such as being a thoughtful mover rather than a stopper, close tracking, and planning ahead of time to break a problem down in useful ways.

Of course, we should also help students carry out such strategies. Teaching a mental model of the computer—that is, the computer as a glass box, not a black box—is helpful (Mayer, 1979, provides suggestions, which will be reviewed

in a future column). The dangers of tinkering could be discussed, and planning strategies such as those suggested in the two previous columns offered as alternatives. A possible side benefit of such teaching is that students will use these new learning strategies in other areas.

Strategies: Final Words

Of course, there are many other ways in which children vary (Watt, 1979). The dichotomies presented in the last few columns (e.g., bottom-up vs. top-down, visual vs. analytic, stoppers vs. movers) are but initial attempts to make sense of children's strategies. Nevertheless, awareness of the various strategies can serve as a framework for figuring out how children are thinking, an important first step in guiding that thinking.

References

Clements, D. H. (1989). What's hard about beginning with Logo? The Research. *Logo Exchange*, *8*(2), 26-30.

Clements, D. H. (1989). Logo: Search and research. To err is human.... *Logo Exchange*, *8*(3), 29-31.

Clements, D. H. (1989). Logo: Search and research. Recurrent recursion misconceptions. *Logo Exchange*, *8*(4), 31-32.

Clements, D. H. (1989). Logo: Search and research. To err is human...to debug, divine. *Logo Exchange*, *9*(4), 32-34.

Clements, D. H. (1990-91). Logo: Search and research. Strategies for solving turtle geometry problems. *Logo Exchange*, *8*(3), 29-31.

Mayer, R. E. (1979). The psychology of how novices learn computer programming. *Computing Surveys*, *13*, 121-141.

Noss, R. (1984). *Children learning Logo programming. Interim report No. 2 of the Chiltern Logo Project.* Hatfield, England: Advisory Unit for Computer Based Education.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York: Basic Books.

Papert, S., Watt, D., diSessa, A., & Weir, S. (1979). *Final report of the Brookline Logo Project. Part II: Project summary and data analysis (Logo Memo No. 53).* Cambridge, MA: Massachusetts Institute of Technology, Artificial Intelligence Laboratory.

Perkins, D. N., Hancock, C., Hobbs, R., Martin, F., & Simmons, R. (1986). Conditions of learning in novice programmers. *Journal of Educational Computing Research*, *2*, 37-55.

Watt, D. (1979). *Final report of the Brookline Logo Project. Part III: Profiles of individual student's work (Logo Memo No. 54).* Unpublished manuscript, Massachusetts Institute of Technology, Artificial Intelligence Laboratory, Cambridge, MA.

Douglas H. Clements is an associate professor at the State University of New York at Buffalo. He has studied the use of Logo environments in developing children's creative, mathematics, metacognitive, problem-solving, and social abilities. Through a National Science Foundation grant, he has co-developed a K-6 elementary geometry curriculum, *Logo Geometry* (published by Silver, Burdett, & Ginn). His most recent book, *Computers in Elementary Mathematics Education*, emphasizing Logo, was published by Prentice–Hall in 1989.

Douglas H. Clements
State University of New York at Buffalo
Department of Learning and Instruction
593 Baldy Hall, Buffalo, NY 14260
CIS: 76136,2027 BITNET: INSDHC@UBVMS

Global Logo Comments

Edited by Dennis Harper
University of the Virgin Islands
St. Thomas, USVI 00802

This month the column will go island hopping to summarize four recent Logo studies. From the island continent of Australia, to New Zealand, to Tasmania, and finally to my home island of St. Thomas in the Caribbean, a brief description and summary of each study will be given along with an address where more details can be obtained.

Frequent *LX* contributor Anne McDougall recently completed a study in Australia that investigated Papert's conjecture that children in a computer-rich learning environment using Logo might be able to engage in activities involving abstract or formal thinking at ages considerably younger than would be expected from Piagetian theory. Her study reported the development of understanding and use of recursion, a topic usually considered difficult for undergraduate computer science students, in children of primary school age.

Recursion was introduced to two children (ages six and nine) using examples of self-reference, repetition with variation, and nesting of levels in pictures and stories. Both children were able to recognize and generate examples of features of recursion in pictures, stories, and everyday situations, although they found it difficult to define or talk in general terms about recursion. The older child, working at first with the researcher and later with a school friend, developed Logo programming skills, including confident use of controlled embedded recursion.

These children, in an environment rich in opportunities for learning about recursion, were able to understand and use this abstract idea when one of them was as young as six years of age. At age nine the elder child was able to read with understanding Logo procedures using embedded recursion, and to prepare with help a procedure containing tail recursion. At age 11 she and a school friend together operated as competent and generally independent programmers with full recursion in Logo. Thus Papert's conjecture that children in a culture rich in resources for building intellectual structures might be able to engage in activities involving abstract or formal thinking at ages younger than would be expected from Piagetian theory is supported, in the case of recursion, by this Australian study.

A New Zealand study conducted by Kwok-Wing Lai of the University of Otago documents a computer based learning environment using LEGO construction elements and the Logo programming language to foster the development of problem solving of young learners. Twenty-four students (aged 10 to 11) participated in this study for eight 90-minute sessions. The cognitive and metacognitive outcomes, the attitudes of the participants towards this LEGO-Logo learning environment, and gender differences were reported.

The LEGO-Logo learning environment investigated in this study was highly motivational to young learners. Both girls and boys were enthusiastic about the project, and no gender difference was observed. The environment was stimulating because the participants were able to control their learning process, and the concreteness of the problem tasks enabled them to relate to what they had encountered before. Results were also clear that this learning environment was conducive to acquisition of knowledge in control technology and skills in problem solving. It also promoted metacognitive awareness.

J.S. and L.L. James of the Tasmanian State Institute of Technology recently completed a three year study using 74 grade 5 and 6 subjects. Children were identified by learning style, and were administered LEGO and Logo activities. The study investigated using these activities to cater for learning style preferences. The results of the statistical analysis do not support the contention that we can identify teaching methods based around LEGO-Logo activities that effectively cater for preferred learning styles, promoting improvement in reason-

ing, and spatial relations test performance. The results of this study tend to encourage the researchers to recommend a mixed-activities approach as one that offers variety and flexibility. Perhaps double open-ended activities can be developed that allow learners to solve problems using both sequential and holistic approaches to problem solving.

Lennox Douglas of St. Kitts/Nevis recently completed a study at the University of the Virgin Islands that once again compared Pascal, BASIC, and Logo. Three classes of 11th and 12th graders were each given one semester of computer applications. In the second semester, the classes split into BASIC, Pascal, and Logo programming. In brief, Douglas found that students' attitudes toward both learning and computers were unaffected by the language used. However, when tested on a standardized test of computer knowledge (including programming constructs and algorithms), the Pascal and Logo groups significantly outperformed the BASIC class.

Any *LX* reader interested in more detailed information concerning these studies should contact the researchers at the following addresses.

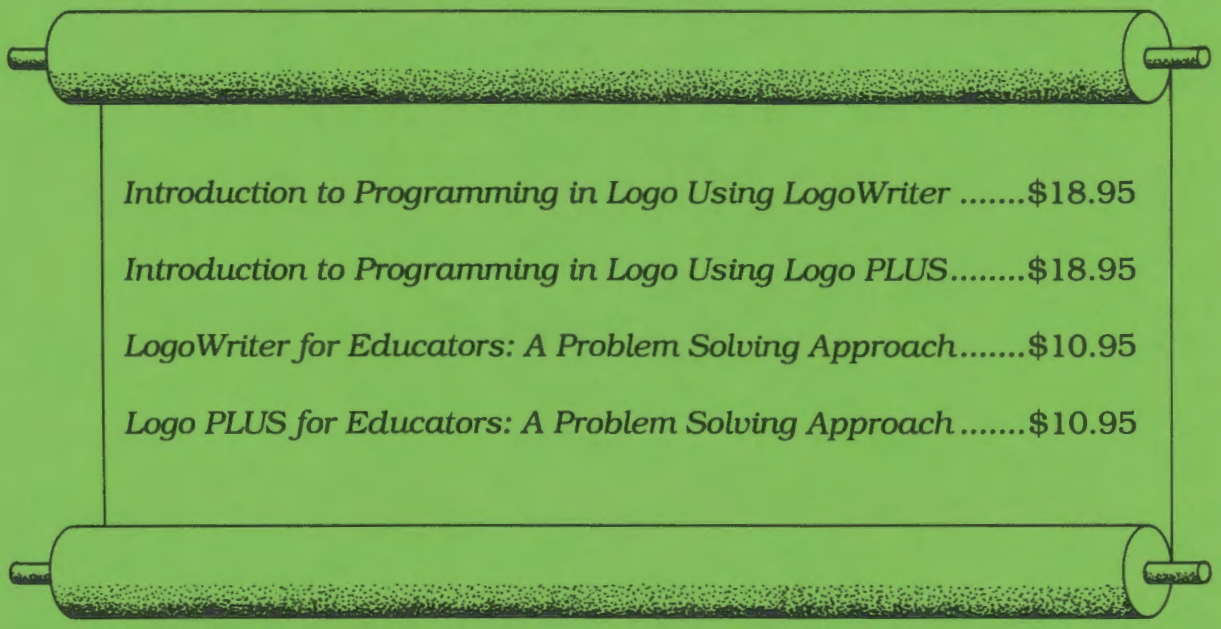
Anne McDougall
Faculty of Education
Monash University
Clayton, Victoria, 3168, Australia

Kwok-Wing Lai
Department of Education
University of Otago
P.O. Box 56
Dunedin, New Zealand

J.S. James
School of Education
Tasmanian State Institute of Technology
Tasmania, Australia

Lennox Douglas
Computer Science
University of the Virgin Islands
St. Thomas, VI 00802, U.S.A.

Look at our Logo list!



Introduction to Programming in Logo Using LogoWriter $18.95

Introduction to Programming in Logo Using Logo PLUS.....$18.95

LogoWriter for Educators: A Problem Solving Approach.....$10.95

Logo PLUS for Educators: A Problem Solving Approach.....$10.95

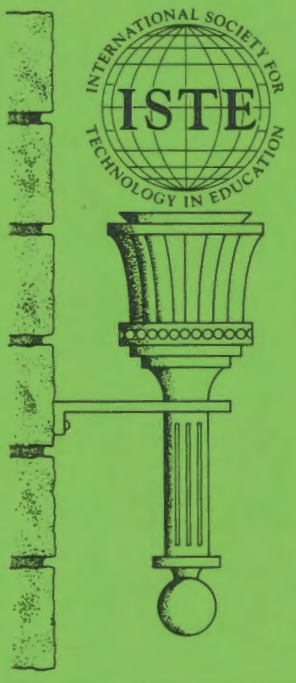
Logo users at all levels benefit from these ISTE selections.

The *Introduction to Programming* books, written by Sharon Yoder, provide beginners with a Logo base to build on and experienced users with a reference to rely on. Both are excellent resources for teacher training or introductory computer science classes.

New from ISTE, *LogoWriter (Logo PLUS) for Educators: A Problem Solving Approach* takes Logo learning to new depths. The focus is entirely on learning and practicing general problem solving skills while using Logo. Great for beginning programming experience. Appendices include keystroke summaries, turtle shape pictures, and a quick reference card. Written by Sharon Yoder and Dave Moursund.

To order, contact: ISTE, University of Oregon, 1787 Agate St., Eugene, OR 97403-9905; ph. 503/346-4414. (Please add $3.25 shipping for single copy orders, $4.50 for up to 4 copies, and $6.00 for 5 copies)

The *International Society for Technology in Education* touches all corners of the world. As the largest international non-profit professional organization serving computer using educators, we are dedicated to the improvement of education through the use and integration of technology.



Drawing from the resources of committed professionals worldwide, ISTE provides information that is always up-to-date, compelling, and relevant to your educational responsibilities. Periodicals, books and courseware, *Special Interest Groups, Independent Study* courses, professional committees, and the Private Sector Council all strive to help enhance the quality of information you receive.

It's a big world, but with the joint efforts of educators like yourself, ISTE brings it closer. Be a part of the international sharing of educational ideas and technology. Join ISTE.

Basic one year membership includes eight issues each of the *Update* newsletter and *The Computing Teacher*, full voting privileges, and a 10% discount off ISTE books and courseware. $36.00

Professional one year membership includes eight issues each of the *Update* newsletter and *The Computing Teacher*, four issues of the *Journal of Research on Computing in Education*, full voting privileges, and a 10% discount off ISTE books and courseware. $69.00

Join today, and discover how ISTE puts you in touch with the world.

ISTE
1787 Agate St., Eugene, OR 97403-1923.
ph. 503/346-4414.

Logo Exchange
ISTE/University of Oregon
1787 Agate Street
Eugene, OR 97403-1923