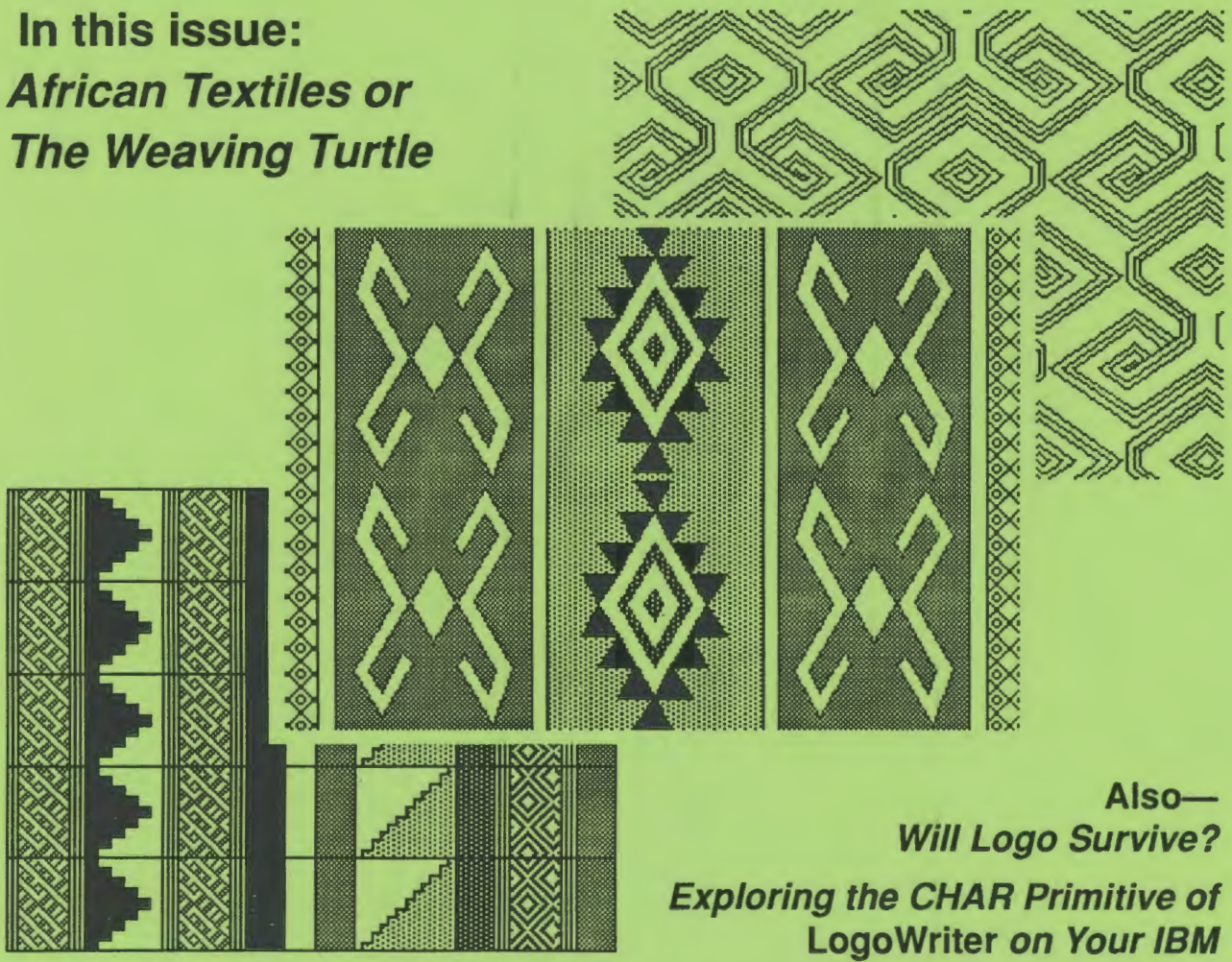
Journal
of the ISTE
Special Interest Group
for Logo-Using
Educators

LOGO EXCHANGE

Winter 1992 Volume 11 Number 2

In this issue:

African Textiles or The Weaving Turtle



Also—

Will Logo Survive?

Exploring the CHAR Primitive of* LogoWriter *on Your IBM

International Society for Technology in Education

Logo Exchange

ISTE • 1787 Agate Street • Eugene, OR 97403-1923 • InterNet: ISTE @ Oregon.uoregon.edu
503/346-4414 • ISTE Order Desk : 800/336-5191

Logo Exchange is the quarterly publication of the International Society for Technology in Education's Special Interest Group for Logo-using educators.

Individual ISTE Membership: $46.00
Dues support the development, coordination, and delivery of ISTE services, including 7 issues of the *ISTE Update* newsletter, either 8 issues of *The Computing Teacher*, four issues of *Educational IRM Quarterly*, or 4 issues of the *Journal of Research on Computing in Education*, full voting privileges, and a 10% discount on ISTE books and courseware. Add $10 for mailing outside the USA.

Individual ISTE Members may receive *LX* for $16.00.
Dues includes a quarterlysubscription to this publication. Add $10 for mailing outside the USA. Send membership dues to ISTE. Add $2.50 for processing if payment does not accompany your dues. VISA, Mastercard and Discover accepted.

LX solicits articles on all topics of interest to Logo-using educators. Please contact ISTE for submission guidelines. Opinions expressed in this publication are those of the authors and do not necessarily represent or reflect the official policy of ISTE.

POSTMASTER: Send address changes to SIGLogo, 1787 Agate Street, Eugene, OR 97403-1923, USA.

ISTE is a non-profit organization with its main offices housed at the University of Oregon.

LOGO EXCHANGE

Volume 11 Number 2 Journal of the ISTE Special Interest Group for Logo-Using Educators Winter 1992

Contents

From the Editor

It's Not as "In" to Be a Logo Advocate as It Once Was

by Hob Brown

So says Sharon Yoder in her editorial in the Summer, 1992 (Vol 10, No 4.) issue of *Logo Exchange*. Even those of us who place Logo only slightly below spouse and children in our affections agree. This decline is all the more mystifying given the spate of outstanding Logo books and articles published in the past few years, as rigorous and thorough as they are pedagogically exciting. Logo should be coming of age, not lapsing into senility. What has happened and, more important, what can be done to reverse it?

Part of the problem is no doubt the cyclical nature of educational novelties that keep our Ph.D. factories in business with new students and new grant proposals. Another part of the problem is that Logo was oversold. Seymour Papert made claims about the positive effects of Logo programming upon general cognitive ability. These claims are still echoed sometimes in *Logo Exchange*, but they are not fully supported by research. (See, for example, the exchanges between Papert [1987] and Roy Pea [1987] concerning research findings that threatened some of these claims.)

But there are other reasons for Logo's decline. Some of these reasons are illustrated by a lesson I used to teach about constructivism, which unwittingly demonstrates teachers' remarkable fascination with, and their rejection of, Logo. Logo is taught in Hong Kong as part of a three-year computer literacy program offered in what is the equivalent of junior high school, yet few teachers besides those in mathematics or computer science have any contact with Logo. So, when I introduce students in my Philosophy of Education class to Logo+*Mindstorms* as a perfect example of constructivist educational thought in action, I can be fairly sure that only a few students are familiar with Logo. Between a third and a half of our inservice students have little or no computer experience at all. (Yes, we still offer a Philosophy of Education class as an option to our teachers in training, and, even more remarkably, our students still sign up for it.)

The Lesson Begins

There are about 25 students. In a front corner of the room is a computer connected to a large television monitor everyone can see. I first use an overhead transparency to briefly introduce basic turtle commands and the rudiments of writing procedures. This transparency is left visible for the rest of the lesson. Using student directions, I then demonstrate a few simple techniques such as making circles, connecting procedures for a stick man, and drawing a spiral. Students then team up in informal groups of two or three and, in turn, do some messing about with Logo for themselves, while I try to explain the ideas behind constructivism to the rest of the class.

From that point on, no one in class pays any attention to me and constructivism whatsoever, while these supposedly reserved and unimaginative university graduates climb all over themselves offering suggestions to those at the computer, laughing at mistakes, planning their own grand creations, and so on. Occasionally I receive a somewhat guilty grin as I continue droning on about the pros and cons of constructivism, because by then all eyes are glued on the computer screen. They simply cannot help it, which is, of course, the whole point of the lesson.

For most students, the point is well made. But then I have to contend with the mathematics and computer science students, a very bright bunch of dissidents. Although they get caught up in the spirit of drawing armies or ever kinkier vortices—the same as the rest—after class I always hear from them a litany of complaints about Logo. They feel that it shouldn't be taught in secondary school and that programming shouldn' t be a part of a computer literacy course. They remark that it's a waste of time to teach Logo rather than C, Pascal, or even structured Basic, and so on. Sure, they admit, Logo is fun for beginners, but that is the problem: once you are no longer a beginner, you need more than Logo has to offer.

Some Suggestions

All right, part of this is simply ignorance of the literature, of what Logo can really do. Part of it may be poor initial instruction in Logo, part is the rage for C and object-oriented programming, and part is a perfectly legitimate skepticism about the suitability of Logo in a computer literacy program. I only wish their skepticism extended to the wisdom of computer literacy courses altogether. But part is also due to the language as we have it today, and how it is understood and presented. I offer the following suggestions as my contribution to rethinking Logo.

1. We should cut Logo's umbilical chord to constructivism once and for all. Used in a certain way, Logo is a marvelous exemplar of constructivism, but I can see no logical or pragmatic necessity, no tangle of inferences, deductions, and causes, that requires us to construe Logo as only operating legitimately in a constructivist framework. If we took this simple and logical step, there would be no need for Mark Horney, in the previously cited issue of *Logo Exchange* (Vol. 10, No. 4), from which the Yoder quotation came, to justify his excellent project because it deviates from the received orthodoxy of what is "normally expected within the confines of the student-centered, exploratory Logo philosophy" (p. 24). The term "confines" puts it very well.

2. We should bring Logo up to date. I use two languages on my Macintosh, *Think Pascal* and *Object Logo*. *Think Pascal* has an editor and debugging facilities that are a joy and a delight. Automatic formatting, automatic syntax checking, and variable tracing are just a few of the aids available. Then I turn to *Object Logo*, which I both respect and enjoy, but in the files window—the only way to write even medium-sized programs—there is no syntax checking, there is no formatting, there is no debugging beyond TRACE and STEP, there is not even a search-and-find or replace utility. These things are expected in any serious language today; and if Logo does not have them, it will not be taken seriously.

3. In a similar vein, I find the Help facilities of Logo primitive, to say the least. Many versions of Basic, Pascal, and C have hypertext access to help continuously available. Symantec publishes a program called *Reference* for use with *Think Pascal* or *Think C*, with explanations and code for virtually all toolbox routines. But turning back to Logo, I find myself once again leafing through the reference manual, sticking in bookmarks, and balancing the manual on my knees while tapping in code. This, in the 1990s, is simply unacceptable.

4. On the other hand, *Object Logo* does permit object programming. It is, in fact, probably the best language around for introducing students to what is meant by terms such as classes or inheritance, as well as the enormous advantages of object programming. One can, of course, mimic objects with any version of Logo, but all complete versions should have at least some object programming capability built in.

5. We should rethink the syntax and control structures of Logo. For example, it is possible to write

```
GODZILLA :cities
   IF :cities = [ ] [ OUTPUT "Chewed ]
   . . . . . .
   GODZILLA BUTFIRST :cities
```

The procedure GODZILLA continues until all cities have been chewed, and then outputs a message to that effect. But the structure of the recursion is obscured by the seeming independence of the conditional line from the lines that follow. The structure of the program is far better shown by:

```
GODZILLA :cities
   TEST EQUALP :n [ ]
   IFTRUE [OUTPUT "Chewed ]
   IFFALSE [···
         GODZILLA BUTFIRST :cities]
```

This notation is possible in Logo, but without some sort of automatic indentation, it is difficult and confusing, particularly if there are further tests in the results of IFFALSE.
If, on the other hand, two successive lines of code are not part of a binary tree, an IF [. . . .] structure works fine and adequately shows what the program is really doing:

```
GODZILLA :cities
   . . . . . .
   IF NOT :cities = [ ] [PRINT
         (SENTENCE "Chewing FIRST
   :cities)]
   . . . . . .
   GODZILLA BUTFIRST :cities
```

Linking the two lines above as they are in the TEST version would be a mistake, since the lines that follow the PRINT (SE . . .) instructions are meant to be executed regardless of the conditional IF test. I don' t see an easy way of making the distinction without using TEST exclusively, or multiplying parentheses as in Lisp, but this ambiguity detracts from Logo's logical rigor.

6. We may want to introduce more parentheses anyway. Putting command after command interspersed with data or variables can produce virtually unreadable code. I personally resort to the liberal use of parentheses and to multiple lines for a single set of connected instructions. If Logo *required* more of this optional grouping, it might make automatic syntax checking much more feasible.

There are undoubtedly many more suggestions others could make, most far better than mine. The point

is that we, the Logo users, need to make these suggestions, and at least some of them need to be implemented if Logo is to retain the use it deserves. There is, perhaps, a trade-off: transforming Logo into a mainline language might undermine some of its simplicity and its direct appeal to young initial users. I don't believe that will necessarily be the case because I don't believe most early users are much concerned with the more complex capabilities of the language anyhow. (How many Logo programmers, even experienced ones, can use THROW or CATCH without consulting the reference manual?) The trick is to keep these complexities out of the way, perhaps even out of sight, until they are needed.

Logo should grow with the user. To do that, it should have some object-oriented capability, a powerful editor, debugging tools for programs that exceed just a handful of procedures, and rigorous, unambiguous control structures. I firmly believe these things can be added without causing grave mental anguish either to primary school children or their teachers. Logo should not have an arbitrary cap, imposed from the days when 64K in an Apple II was all there was. It should not have a ceiling that makes budding young programmers—or their teachers—give up on it as kid stuff.

References

Papert, S. (1987). Information technology and education. *Educational Researcher, 16*(1), 22-30.

Pea, R. D. (1987). The aims of software criticism: Reply to Professor Papert. *Educational Researcher, 16*(5), 4-8.

Hob Brown
Department of Education
University of Hong Kong

Pictures in This Issue

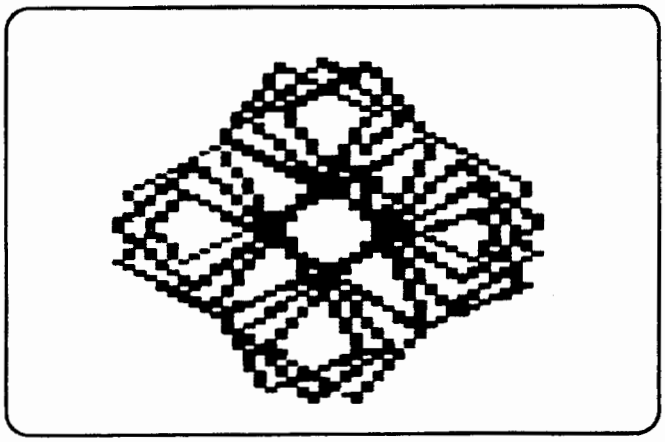
by Jandy Bird

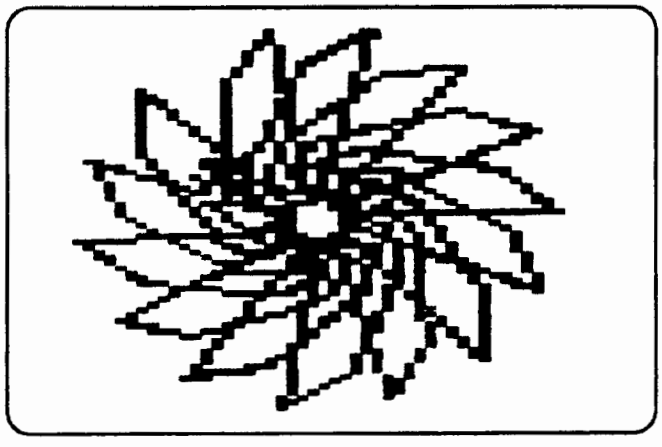
Students at Conover Road School, a K-4 elementary school, have computer lab activities centered around *LogoWriter*. Each class in grades 1-4 has a lab session once every other week with Mrs. Arlene Radman, computer teacher. The classroom teacher assists in the lesson, and lessons are often planned to complement the regular classroom program.

In addition, students have a computer in their individual classrooms at least three days a week. They work at the teacher's direction on *LogoWriter* or on activities using other software applications. Each class has a volunteer computer parent who attends lab with the class and who often works individually with students. Kindergartners have in-class lessons with Mrs. Radman, and parent helpers assist with hands-on activities.

Students in the gifted/talented resource room program have access to Apple II and Macintosh computers, since they share the physical space with the computer lab. Working with Dr. Jandy Bird, they use *LogoWriter*, Lego/Logo, and other applications to develop individual and group projects out of their own interests and to explore the possibilities of using technology to express their ideas.

Jandy Bird
Conover Road School
Colts Neck, NJ 07722
CompuServe: 73517.3270





Tony Bean, a second grader, did a series of design patterns using rotations.

Velcro Logo

by Tom Lough

Recently, a workshop speaker told a story that really got me thinking. He was a avid baseball fan, and he hoped fervently that his nine-year-old son would grow up to be a professional baseball player.

When his son was seven, he had decided it was time to introduce him to the fundamentals of catching and throwing. He went to the local sporting goods store and bought his son the biggest and the finest first-base glove in stock. How proud he was when he presented it to his son!

Unfortunately, the glove was much too big and heavy for the young boy's hand. After several clumsy unsuccessful attempts, the dad ended up taping the glove on the boy's hand in desperation. (He said this was a true story.)

As you can imagine, all of the son's attempts to catch a ball ended in disaster. Frustration and tears soon followed. Then shouting and screaming. This certainly did not create a favorable attitude toward baseball.

Fortunately, the dad had the good sense to try again. He went to a toy store and found a smaller glove, one about the size of a young boy's hand. Moreover, the inside of the glove and the outside of the ball that came with it were covered with Velcro.

Needless to say, the boy was happy to get a small glove for his small hand. Moreover, whenever he moved the glove anywhere remotely near the ball, the Velcro snagged it. He couldn't miss!

His attitude toward baseball improved remarkably. This was fun! He could catch the ball! He was succeeding! Later on, he was able to meet the challenge of catching with a non-Velcro glove and ball with no problem.

What does this have to do with Logo? For me, it was a hard-hitting reminder. I have to remember that just because I have known Logo for several years, the people I am teaching are just beginning. If I expect them to write procedures or use recursion or conditional statements right off the bat (sorry, I couldn't resist), I am asking for frustration and tears.

FORWARD and BACK, LEFT and RIGHT.

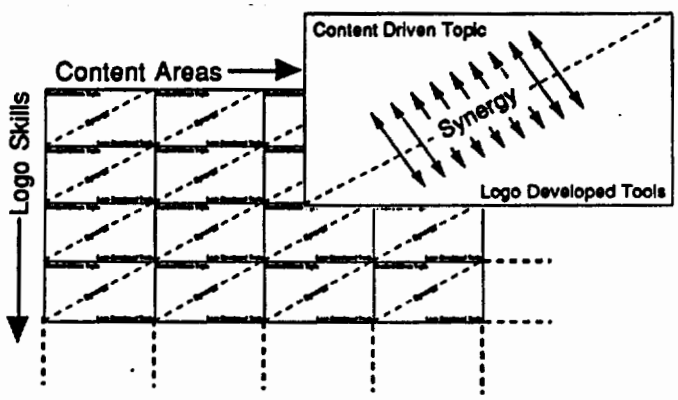
These are the Velcro commands of Logo. With them we can succeed. We can have fun. We can prepare effectively for the more challenging levels of this engaging and powerful programming language.

One of my favorite things is a song that opens with a line I do not let myself forget. "Let's start at the very beginning, a very good place to start." That is what Velcro Logo is about. A very good place to start, indeed.

FD 100!

Tom Lough
Founding Editor
PO Box 394
Simsbury, CT 06070

Oops!



In the Fall 1992 issue of *Logo Exchange*, a graphic in Mark Horney's *Logo Across the Curriculum* was inadvertantly scrambled.

A corrected version of the graphic appears at left.

Also in that issue's Logo Ideas Column, a block of code was incorrectly given. The procedure results should read:

```
to results
clearpage
label.table
display 25
```

Berkeley Logo Available

An alpha test release (version 2.8) of Berkeley Logo is now available by anonymous FTP. This version fixes several important bugs (the one about spaces at the end of a line, for example). Its major new feature is an extended-memory capability for the PC.

FTP to anarres.cs.berkeley.edu and get the following files:

pub/logo.tar.Z	Unix sources and documentation
pub/ucblogo.zip	PC version, PKZIP form, including executable BL.EXE
pub/ucblogo.sit.hqx	Mac version, StuffIt/ BinHex form, with executable Logo
pub/usermanual	Just the User's Manual
pub/csls.tar.Z	Logo programs from *Computer Science Logo Style*

You're on your own for the unpacking software.

Advantages of Berkeley Logo:

- It's free.
- It comes with source files (in C).
- Logo programs are completely compatible among Unix, PC, and Mac.

Disadvantages of Berkeley Logo:

- It's pretty slow.
- It doesn't do anything fancy about graphics (one turtle).

If you don't have FTP access, but can read a 3.5-inch disk, send me $2.50 to cover costs and tell me whether you want the Mac or the PC version (or send $5.00 for both) and I'll send you a disk. (Outside North America, please send US $3.50 for one disk or $6.50 for both.) The PC disk includes both PC versions.

Brian Harvey 2634 Virginia St.
Berkeley, CA 94709
bh@anarres.CS.Berkeley.EDU

Some Logo Trivia

by Paul M. Wexelblat

1. Logo was named by Wally Feurzeig of BBN. It is loosely connected with the Greek *logos*, meaning word. (Logo is not an acronym and should not be all capitals.)
2. The first turtle, an innovation with which Seymour Papert was heavily involved, was a radio-controlled rolling robot named "Irving."
3. The original name of Logo was "Ghost" and dates from 1966 It was implemented by Dan Bobrow in Lisp on a PDP-1 at BBN.
4. An interesting quick history of Logo may be found in Wally Feurzeig's article "The Logo Lineage," which appears in *DITITAL DELI* (1984), edited by Steve Ditlea (New York: Workman Publishing), ISBN 0-89480-591-6.

Paul Wexelblat
Dept of Computer Science
University of Massachusetts—Lowell
One University Ave
Lowell, MA 01854
508/934-3648
wex@cs.ulowell.edu

Flags of the World

by Dorothy Fitch

The flags of the world abound with striking colors and fascinating patterns. If you take a closer look, you'll find that most share similar design elements. They use the same building blocks for their design—stars and stripes, of course, but also bars, circles, squares, and triangles.

In this column, we'll draw some of these flags and create tools to make the job easier. We can use our flag tool box to draw many existing flags as well as those of your own invention!

Beginners will find that drawing some flags is quite easy. Just type a few instructions, add a touch of color, and you're done. Experienced users will want to try more complicated designs and perhaps make tools that create variable size flags. The projects listed at the end will give you some more ideas.

Disclaimer: Because of the state of global politics, there is no guarantee that these flags are currently those of the countries listed, nor is there any guarantee that these countries even currently exist! My "official" source for these flag designs is a colorful T-shirt from Walt Disney World labeled "The Colors of Our World—An International Celebration."

Getting Started

Take a look at the designs on the Flag Page at the end of this article. What do they all have in common? Their rectangular shape or outline. We'll start by writing a procedure that draws a rectangle.

```
TO RECTANGLE
REPEAT 2 [FORWARD 120 RIGHT 90 FORWARD
   180 RIGHT 90]
END
```

Try it out. Do you like where the rectangle is drawn on the screen? If not, write a procedure that sends the turtle to a better starting place. We can also use this procedure to return the turtle to that spot.

```
TO H
PENUP
SETX -60
SETY -40
PENDOWN
END
```

The procedure to begin any flag design might then look like this:

```
TO OUTLINE
DRAW  ; or CG
H
RECTANGLE
END
```

What might be some other useful tools? Many of these flags use stripes (short, wide rectangles) and bars (tall, narrow rectangles).

Flags (including some not pictured here) may have as few as 2 stripes or as many as 13 stripes. It would be handy to have one stripe procedure that can draw a rectangle that is 1/2, or 1/3, or 1/13 the height of the flag. This procedure uses a variable to determine how tall to make the stripe.

```
TO STRIPE :AMOUNT
REPEAT 2 [FORWARD 120 * :AMOUNT RIGHT
   90 FORWARD 180 RIGHT 90]
END
```

To draw a stripe that is half the height of the flag, type STRIPE 1/2. To draw one that is 1/3 the height of the flag, type STRIPE 1/3. For the USA flag, you would type STRIPE 1/13.

A procedure to draw Austria's flag, which has three stripes, would then look like this:

```
TO AUSTRIA
OUTLINE
STRIPE 1/3
FORWARD 120 * 1/3 ; or 40
STRIPE 1/3
H
END
```

It is interesting that to end up with three stripes, we could also type this:

```
OUTLINE
STRIPE 1/3
STRIPE 2/3
```

Using this method to get all the stripes in place isn't an obvious choice, but it works nicely. It is also easier than

figuring out how far to move the turtle forward to get to the next stripe.

Gee, have you noticed how we're using fractions! This might be a good way to introduce fractions to young people. They can add stripes to flags by counting how many parts of the whole each stripe should represent.

To draw flags that are divided vertically, we can write a similar BAR procedure.

```
TO BAR :AMOUNT
REPEAT 2 [FORWARD 120 RIGHT 90
   FORWARD 180 * :AMOUNT RIGHT 90]
END
```

So, a procedure to draw Italy's flag would look like this:

```
TO ITALY
OUTLINE
BAR 1/3
BAR 2/3
H
END
```

Notice that we haven't dealt with color. We'll leave it to you to fill the stripes and bars with colors that are appropriate for each country's flag. If your machine or version of Logo does not support color, print out the flag and color it off-line.

More Tools

A Circle Tool

Some of the flags have other elements we can easily draw in Logo. For example, the Japanese flag has a circle in the center. You could draw a circle (or, more precisely, a multi-sided polygon) in standard Logo style, like this:

```
REPEAT 360 [FORWARD 1 RIGHT 1]
```

Adjust the size of the circle by changing the number after FORWARD. The trickiest part of using this method is finding the right place in the flag outline to begin drawing the circle.

You might want to consider drawing the circle using a different method—one where the turtle is based at the center of the circle.

```
REPEAT 360 [PENUP FORWARD 20 PENDOWN
   FORWARD 1 PENUP BACK 21 RIGHT 1]
```

Adjust the size of this circle by changing the number after the first FORWARD command. (Don't forget to make the number after BACK one larger!) It is easier to find the middle of the flag than to find the correct starting point for the other type of circle.

Here is a procedure for drawing the Japanese flag design:

```
TO JAPAN
OUTLINE
PENUP
RIGHT 90
FORWARD 90
LEFT 90
FORWARD 60
PENDOWN
HIDETURTLE
REPEAT 360 [PENUP FORWARD 24 PENDOWN
   FORWARD 1 PENUP BACK 25 RIGHT 1]
H
END
```

A Star Tool

You may have drawn a Logo star that looks like this

using these instructions:

```
RIGHT 18
REPEAT 5 [FORWARD 25 RIGHT 144]
LEFT 18
```

But most flags have a solid star, with no lines inside, like this:

So our challenge is to draw this star. There are a couple of approaches we could take. Have you noticed the shape formed by the lines inside the first star above? The lines make a pentagon.

One approach would be to draw the star using the instructions above, then erase the lines of the pentagon. This is easier said than done, as you will find out if you try. My attempts left remnants of the pentagon lines on the screen.

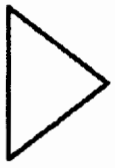
A second approach would be to move the turtle around the outside of the star shape. This sounds more complicated than it actually is, though it is still a bit of a project. Using trial and error, I wrote a procedure to draw the two lines that define a point of the star. Then I drew this

point to the left of each side of a pentagon. After figuring out the distances using actual numbers, I replaced them with a variable so that I could draw a solid star of any size. Here are the procedures:

```
TO STAR :SIZE
RIGHT 18
REPEAT 5 [POINT :SIZE PENUP
   FORWARD :SIZE RIGHT 72 PENDOWN]
LEFT 18
END

TO POINT :SIZE
PENUP
FORWARD :SIZE
LEFT 108
PENDOWN
FORWARD :SIZE * 1.625
LEFT 144
FORWARD :SIZE * 1.625
LEFT 108
END
```

A Triangle Tool

The triangle in the flag of Sudan poses another interesting challenge. It is not an equilateral triangle. If all the sides were the same length, the shape would extend too far into the flag outline. Here is the triangle I wanted:



Again using trial and error, I hit upon these instructions, which create the correctly shaped triangle:

```
TO TRIANGLE
FORWARD 120
RIGHT 128
FORWARD 97.5
RIGHT 104
FORWARD 97.5
RIGHT 128
END
```

Squares and More Rectangles

The flags of the Scandinavian countries all share the same cross design. When I first drew this flag in Logo, I moved the turtle forward, right, and left to draw the individual lines around the shape of the cross. But reflecting on the design later, I realized that it might have been easier to draw two squares and two rectangles, one shape starting at each corner of the flag outline.

Finding more than one solution to a Logo problem has always fascinated me. Perhaps you will come up with a way to draw this flag that is even easier or more efficient!

Flag Tool Projects

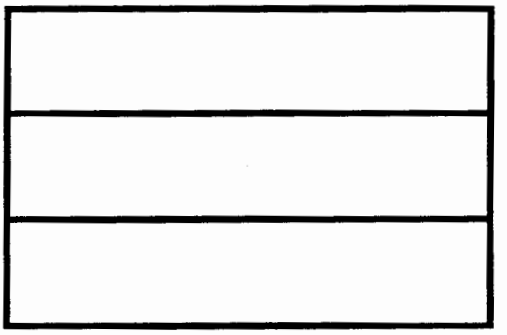
Now you have a powerful set of flag-making tools—stripes, bars, circles, stars, triangles, and squares. Here are some ideas for classroom projects.

- There are many other countries whose flags use these same elements. Research the design for the flags for these countries and territories and use your Logo tools to draw them.
  Bangladesh, Benin, Botswana, Bulgaria, Burkina-Faso, Cameroon, Central African Republic, Colombia, Cuba, Czechoslovakia, Djibouti, Dominican Republic, Dutch West Indies, England, Gambia, Ghana, Greece, Greenland, Guinea-Bissau, Honduras, Iceland, Indonesia, Laos, Liberia, Madagascar, Mauritius, Micronesia, Monaco, Morocco, Mozambique, Niger, North Korea, Norway, Oman, Palau, Poland, Puerto Rico, San Marino, Sao Tome and Principe, Senegal, Solomon Islands, Somalia, Surinam, Switzerland, Thailand, Tonga, USA, Venezuela, Vietnam, West Samoa, Yugoslavia.
- Invent a flag for an imaginary country.
- Invent a flag for a newly formed country. (Check the daily news for the latest offerings!)
- Make a scrapbook or bulletin board display of all your flag designs.
- Find the geographic location of the countries for which you have drawn flags. Use a string to connect each flag to its country's location on a map in a display.
- Print flag designs that younger children can color. Give them a color picture of the flag to use as a guide.
- Draw a mirror image of your flag and print out both designs. Cut them out and paste them together, back to back. Tape your flag to a thin dowel. Make a semi-circle of flags on their flagpoles, just as you would see at the United Nations Building in New York City.
- Make the OUTLINE procedure take a variable for its size. Its width should be 1.5 times its height. Rewrite your flag procedures so that you can draw one in any size you want.
- Add the name of the country to the screen showing its flag, then save the screen as a picture. Write a procedure to present a slide show of many flags.

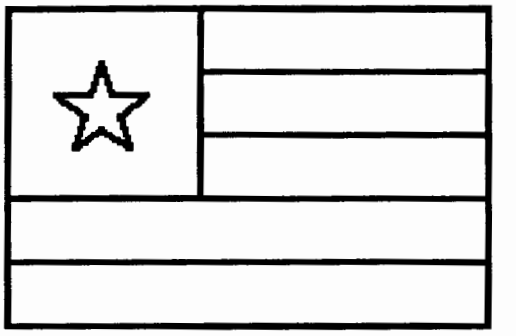
That's all for now. Happy Logo adventures!

Dorothy Fitch
Terrapin Software, Inc. 400 Riverside Street
Portland, ME 04103
CompuServe: 71760,366
Internet: 71760.366@COMPUSERVE.COM

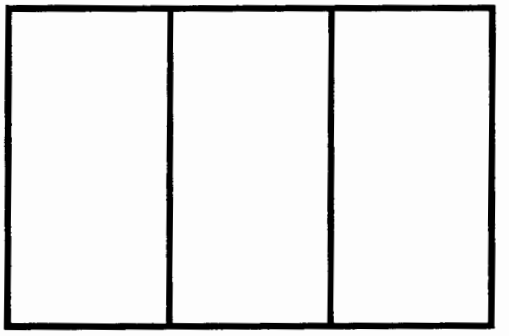
Copy Me Flag Page



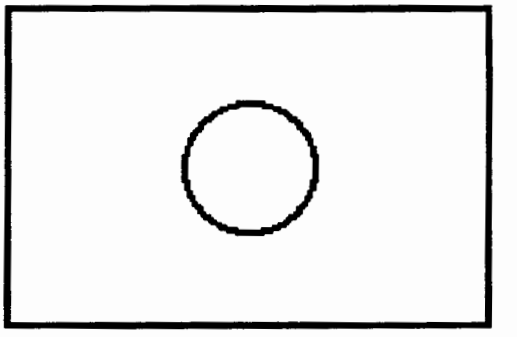
Austria, Bolivia, Netherlands, Luxembourg, Hungary, Sierra Leone, Gabon, BRD, Ethiopia, Upper Volta



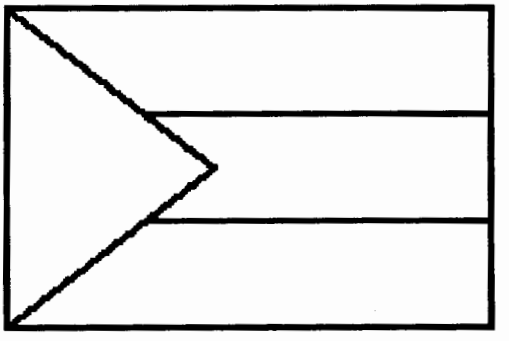
Togo



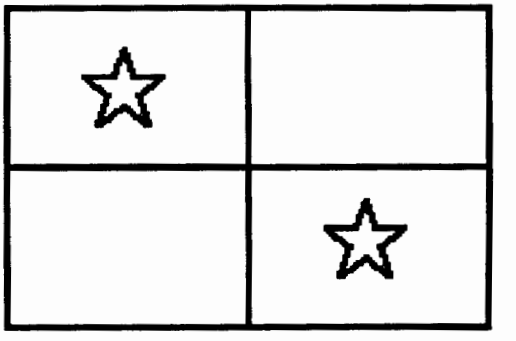
Italy, Belgium, Guinea, Guatemala, France, Chad, Ivory Coast, Mali, Nigeria, Ireland



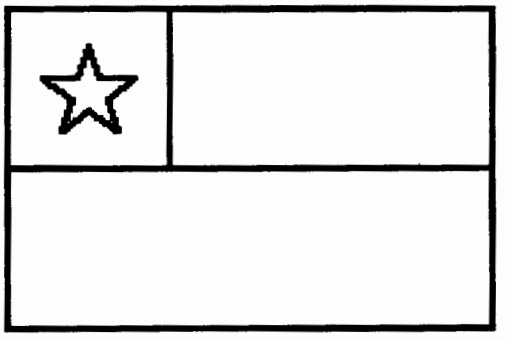
Japan



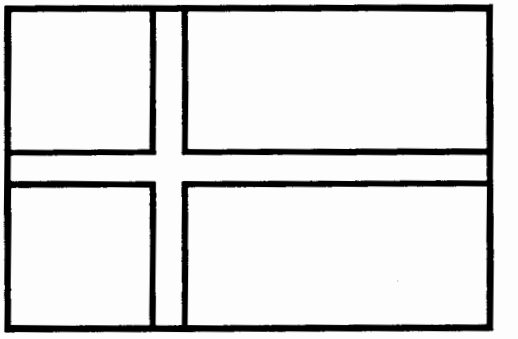
Sudan, Bahamas



Panama



Chile



Finland, Denmark, Sweden

It's About Time!

by Eadie Adamson

A few years ago I worked with a group of boys writing game programs. We were using *LogoWriter* for Apple computers. The boys really wanted a way to have a clock or timer in their games. At that time, there wasn't a good way to do that. Instead, we built counters that kept track of how many times someone "shot" or how many times one object touched another or crossed a given point.

Clock Substitutes

To do this, we set up a starting variable.

```
make "times 0
```

Each time the score or counter was incremented, a little procedure in the program went to work.

```
to addit
make "times :times + 1
end
```

In some cases, the procedure also determined if a maximum value had been reached. The **addit** procedure was then modified to stop the game.

```
to addit
make "times :times + 1
if :times > 10 [print [Game Over!!]
   stopall]
end
```

Back to Clocks

In spite of our alternative solution to the timing problem, the clock problem continued to be intriguing. We played around with simulating them graphically, following some suggestions on a *LogoWriter* activity card. One student spent weeks changing the amount of a **wait** command to various decimals in order to make his clock truly accurate. Then he'd set it going and watch it run through 5 or 10 minutes. I had never before seen this student so actively involved in problem solving. He would come by to talk to me about the problem between classes, bursting with enthusiasm about some new insight into how time was registered.

At one point I began thinking about creating a digital clock, using the label command rather than printing, so that I could put the clock anywhere I wanted on the screen. (Some versions of Logo let you specify a cursor position, but in *LogoWriter* you must move the cursor down and across the screen to put a clock on the lower right side of the screen. Using **label**—text as graphic—avoids the problem.)

Thinking about a digital clock is interesting. I found myself exploring all sorts of problems with synchronization. I quickly realized that this kind of Logo clock couldn't really be accurate because "garbage collection" occurs from time to time. Garbage collection occurs when you see the turtle pause briefly in the midst of a long or complex graphic procedure. Even planning ahead and forcing garbage collection at the beginning of a procedure doesn't really help. (In *LogoWriter*, the primitive that forces garbage collection is **recycle**. Many other versions of Logo have a similar function. See your technical manual for more information.)

Before You Program, Try to Define the Problem!

To create a digital clock, you must first think about building a counter that starts at 0 and goes to 59. When it has registered 59 it turns back to zero. That will do for counting minutes and/or seconds. The hour problem involves counting from 1 to 12 (or 24 if you're measuring time in military or European style). Each time a second passes 59, a minute should move up one. Each time the minutes pass 59, the hour should move up one.

There are some other small problems in setting up a digital clock. When the seconds or minutes are a single digit, a digital clock prefaces the number with a 0. In Logo you need to make sure that when there's only one number, a zero is placed before it. You will need to write what Michael Tempel of the Logo Foundation calls a "pad" procedure that checks if an input has two digits and, if not, adds a 0.

It seems clear that the procedure needs to keep testing a number of values and resetting them when necessary. To avoid having to move the cursor line by line and column by column, I decided to use the **label** command so that I could put the clock anywhere on the screen.

Remember, **label** puts text to the right of the turtle. A clock created by using **label** can be placed anywhere on the screen simply by moving the turtle and then hiding the turtle. One way to use **label** is to use it as if you were creating a flashing sign. Label once and the text appears; label the same text again and it disappears. (I could also have used **clean**, which erases all the graphics but does not move the turtle. **Clean** doesn't

work well if it is inserted into a game or story that includes graphics!)

Here's my first try at a digital clock.

```
to time :hour :minute
if :minute = 60 [name :hour + 1 "hour
   name word 0 0 "minute if :hour >
   12 [name word 0 1 "hour] ]
if equal? 1 (count :minute) [name word
   0 :minute "minute]
if equal? 1 (count :hour) [name word 0
   :hour "hour]
repeat 2 [label (word :hour ":
   :minute) wait 10 ]
time :hour :minute + 1
end
```

I added a **wait** between each **label** command. Just as my student did with his graphic clock, I could then adjust the length of the **wait** to try to come up with a reasonable simulation of time. I also decided, for practical reasons, to allow time to speed up a bit when counting only hours and minutes. If you prefer, it can be adjusted to print instead (take out the **repeat**). Can you then add the seconds?

Once you have solved the basic clock problem, here are some other ideas to consider:

- Build a translator for someone used to European 24-hour time, so that users can plug in 10:00 p.m. and get back 22:00 (or the reverse)
- Make a printed question/answer display or use your digital clock to show users what time it is.
- Build a pair of clocks and have them run together, one on a 12-hour cycle and one on a 24-hour cycle

More About Clocks

My students and I have done a lot of experiments using GS or Macintosh computers. The Apple IIGS computer has a clock that can be accessed with some versions of Logo, including *Logo Express*. This clock gives time to the nearest minute. The Macintosh clock registers seconds as well. Most versions of Logo for the Macintosh can access the clock. Check the manual to see how your version does it. Many MS-DOS computers also have a clock.

It's fun to set up a long procedure—an overnight experiment, perhaps—and print the time at the beginning and at the end. You can also make a control key in *LogoWriter* versions that will print the time when you press the key. Last year one of my Logo math students, a young fourth grader, ran several overnight or weekend experiments. Being able to have the time print was extremely useful, especially when he miscalculated on a project that was to run all weekend and instead ran for only about 14 hours!

Here are some ideas you can work on with a version of Logo that can access the computer's clock.

- Write a seconds procedure that takes the computer clock and reports only the seconds. Write another for minutes only—and one for the hour only. Write one for minutes and seconds.
- Write a procedure that shows elapsed time, subtracting the finish time from the start time.

Suppose you don't have a clock, but you are using *Lego Logo* or *LogoWriter Robotics*. These versions of Logo, as well as several others, contain timers. You might try simulating a clock using the timer. The timer reports 10ths of a second. You'll need to divide the timer's result by 10 to get the seconds. And for minutes?

Time's up! Send me your solutions and suggestions for other problems. I'll feature them in a subsequent article about time.

Eadie Adamson
1199 Park Avenue - 3A New York, NY 10128
212/876-3276
LogoExpress: EadieA CompuServe: 73330, 3266

Statement of Ownership, Management and Circulation (Required by 39 U.S.C. 3685). 1A. Title of Publication, **Logo Exchange** B. Publication No., 08886970. 2. Date of filing, September 17, 1991. 3. Frequency of issue, quarterly. 3A. No. of issues published annually, 4. B. Annual subscription price, $29 in U.S. 4. Complete mailing address of known office of publication (not printer), ISTE, 1787 Agate St., Eugene, OR 97403-1923. 5. Complete mailing address of the headquarters of general business offices of the publisher (not printer), ISTE, 1787 Agate St., Eugene, OR 97403-1923. 6. Full names and complete mailing addresses of the publisher, editor, and managing editor, Publisher, ISTE; Editor, Sharon Yoder; Managing Editor, David Moursund; 1787 Agate St., Eugene, OR 97403-9905. 7. Owner, International Society for Technology in Education, 1787 Agate St., Eugene, OR 97403-1923. 8. Known bondholders, mortgages, and other security holders owning or holding one percent or more of total amount, None. 9. The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes has not changed during preceding 12 months. 10. Extent and Nature of Circulation. Average No. Copies Each Issue During Preceding 12 Months. A. Total no. copies (net press run), 1,500. B1. Paid Circulation, sales through dealers and carriers, street vendors and counter sales, 0. B2. Paid Circulation, mail subscriptions, 1,200. C. Total Paid Circulation (sum of 10B1 and 10B2), 1,320. D. Free distribution by mail, carrier or other means, samples, complimentary, and other free copies, 50. E. Total Distribution (Sum of C and D), 1,370. F1. Copies not distributed, office use, left over, unaccounted, spoiled after printing, 130. F2. Copies not distributed, returns from news agents, 0. G. Total (Sum of E, F1 and 2—should be equal to net press run shown in A), 1,500. Actual No. Copies of Single Issue Published Nearest to Filing Date. A. Total no. copies (net press run): 1,460. B1. Paid Circulation, sales through dealers and carriers, street vendors and counter sales, 0. B2. Paid Circulation, mail subscriptions, 1,269. C. Total Paid Circulation (sum of 10B1 and 10B2), 1,269. D. Free distribution by mail, carrier or other means, samples, complimentary, and other free copies, 51. E. Total Distribution (Sum of C and D), 1,320. F1. Copies not distributed, office use, left over, unaccounted, spoiled after printing, 140. F2. Copies not distributed, returns from news agents, 0. G. Total (Sum of E, F1 and 2—should be equal to net press run shown in A), 1,460. 11. I certify that the statements made by me above are correct and complete. Patti Van Ordstrand, Distribution/Circulation Manager, ISTE.

Tell Me in English—Not Logo!

by Donna Rosenberg

In the past, whenever I presented a specific Logo project to my students and asked them to tell me what must be done, the few responses would consist of hesitantly given Logo commands.

Because most of the children did not know the exact angle to turn the turtle or exactly how far forward or backward the turtle should move, they would not volunteer a response. This tendency to think they should answer in Logo resulted in many of my third- and fourth-grade students believing that they did not know how even to begin work on the project or that they were somehow not good enough at Logo. These same children enjoyed experimenting with Logo and discovering what Logo could do. Yet when asked to create a specific design, they would think it was too hard.

I questioned why this was happening. Was the problem that the children were trying to think exclusively in Logo? Could they complete a project if they thought out the entire process in English first? And how could I facilitate this new way of thinking about projects. The following is a discussion of the process I went through to answer these questions and what I learned during that process.

Tell Me in English.

While working with a third-grade class on how to write a simple procedure, I requested that they make that old standby, a house. My purpose was twofold: to teach them how to write a procedure and to emphasize that Logo is a computer language.

I asked them to give me directions on how to make a simple house. I emphasized that I wanted these

directions in English. Whenever I received an answer that was in Logo (i.e., FORWARD 50), I would reemphasize that I wanted them to tell me in English exactly what they wanted the turtle to do. An acceptable answer would be "move the turtle forward, or move the turtle up, or turn the turtle." Using the chalkboard, I followed each direction as it was given and also wrote the directions under a heading labeled "English." If a direction included a word that was also a Logo primitive, I capitalized that word. When we finished, the chalkboard looked like this:

ENGLISH:
Make a Square
Move the turtle to the top of the Square
Turn the turtle 30 degrees RIGHT
Make a Triangle
Turn the turtle LEFT so it is headed straight up
Move the turtle BACK where it started.

After eliciting the English directions, I asked the class to give me those same directions in Logo, using the English directions while giving Logo commands. While I worked at the chalkboard on one Logo command at a time, all children entered the commands at their computer (in the immediate mode).

The next step was to put this list of commands into a procedure. First I asked for the Logo word (primitive) to make a procedure (TO), and then we discussed what would be a good name for this procedure—something that would help us remember what the procedure does. We also discussed how to end a procedure. I then added "TO HOUSE" to the top of my Logo list of commands and "END" to the bottom of my Logo list of commands. When we finished, the chalkboard looked like this:

ENGLISH	LOGO
to make a house	`TO HOUSE`
make a square	`REPEAT [FORWARD 50 RIGHT 90]`
move the turtle to the top of the square	`FORWARD 50`
turn the turtle 30 degrees RIGHT	`RIGHT 30`
make a triangle	`REPEAT 3 [FORWARD 50 RIGHT 120]`
turn the turtle straight up	`LEFT 30`
move the turtle back where it started	`BACK 50`
the end	`END`

The class entered this procedure into their workspace and then executed it. During the remainder of class time, they experimented with this shape. They spun the house shape, made a row of houses, used color, and/or filled the house(s) with color. With the exception of the HOUSE procedure, they did everything in the immediate mode.

Misconception Corrected

A common misconception the children exhibited was that the "turn" command in Logo not only causes the turtle to turn but that the turtle continues moving. When using the "Tell Me in English" approach, whenever "turn the turtle" was used, it was immediately followed by another command (i.e., move the turtle back or make a square), thereby emphasizing the fact that the turtle had NOT MOVED from its present position. At the same time, it stressed that the "turn" (RIGHT or LEFT) was based on the turtle's orientation.

From Simple to More Complex

I also found the "Tell Me In English" approach to be successful with fourth-grade students involved in more complex problems. When they were working on the use of procedure inputs, I requested that they too make a simple house. After discussing the two shapes they would need to make this house (SQUARE and TRIANGLE), I had the children write a procedure for each.

```
TO SQUARE
REPEAT 4[FORWARD 50 RIGHT 90]
END

TO TRIANGLE
REPEAT 3[FORWARDD 50 RIGHT 120]
END
```

I then asked them to give me directions on how to make this simple house, emphasizing that I wanted the directions in English.

After eliciting the English directions, I asked the class to give me those same directions in Logo. They worked at their computers while I wrote on the chalkboard. When we finished, the chalkboard looked like this:

ENGLISH	LOGO
to make a house	`TO HOUSE`
make a square	`SQUARE`
move the turtle to the top of the square	`FORWARD 50`
turn the turtle 30 degrees RIGHT	`RIGHT 30`
make a triangle	`TRIANGLE`
turn the turtle straight up	`LEFT 30`
move the turtle back where it started	`BACK 50`
the end	`END`

Next I asked the children to change the size or length of their house to 20. Once they successfully completed this task, I asked them to add an input to their procedure that would make a house any size or any length they wanted.

They immediately changed the SQUARE, TRIANGLE, and HOUSE procedures to include the procedure input SOMESIZE rather than a constant number, for example:

```
TO SQUARE :SOMESIZE
REPEAT 4[FORWARD :SOMESIZE RT 90]
END
```

Their finished procedure looked like:

```
TO HOUSE :SOMESIZE
SQUARE :SOMESIZE
FORWARD :SOMESIZE
RIGHT 30
TRIANGLE :SOMESIZE
LEFT 30
BACK :SOMESIZE
END
```

The most common problem I encountered with their work on this project was that the children did not use the procedure input SOMESIZE for both the FORWARD and BACK commands. One student used the procedure input in the turn command (RIGHT :SOMESIZE or LEFT :SOMESIZE). However, most of the children were able to make the necessary changes and successfully complete this project with little or no help from me.

The children who finished used the remaining class time to expand upon this project; they made a town consisting of different sized houses.

Conclusion

Because this method worked with bite-sized pictures (make a square or make a house), third-grade students were not reluctant to use the REPEAT command; nor, for that matter, were fourth-grade students hesitant to use another procedure (i.e., TO SQUARE, TO HOUSE) or to undertake more complex projects (i.e., TO TOWN).

I've found that the "Tell Me In English" approach has enhanced the effectiveness of the Logo teaching and learning process in my class in several ways. It is a wonderful way to teach procedure writing, both for simple and complex procedures. It clarifies and simplifies procedure writing by allowing the children to think through an entire project in their own language before attempting to write it in Logo. It allows children to concentrate on what the turtle should do because they do not have to simultaneously translate their thinking into Logo. By separating the process of describing what the turtle should do and putting these ideas into the Logo language, it emphasizes that Logo is a computer language. The "Tell Me in English" approach has proven to be an excellent method of teaching Logo in my classroom.

Bibliography

Cohen, R. (1990). Logo in the primary classroom: Should simplified versions be used? *The Computing Teacher, 17*(7), 41-43.

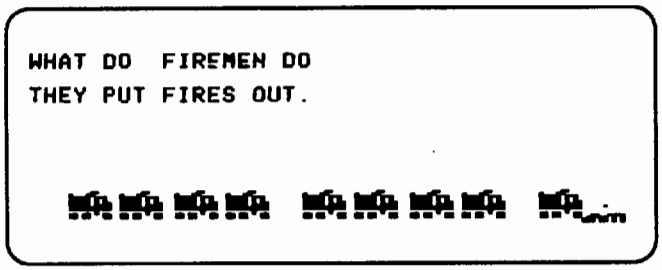
Clements, D. (1989). Learning and teaching Logo problem solving: A summary. *Logo Exchange, 8*(1), 28-29

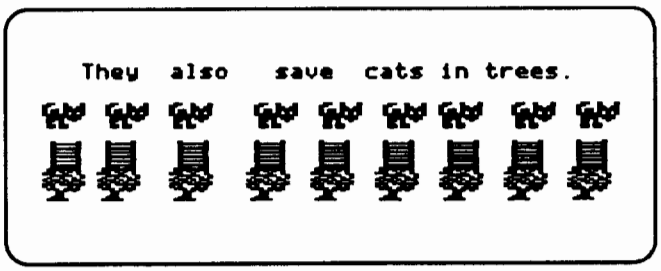
Watt, M. (1989). When teacher and student are stumped. *The Computing Teacher, 16*(8), 30-32.

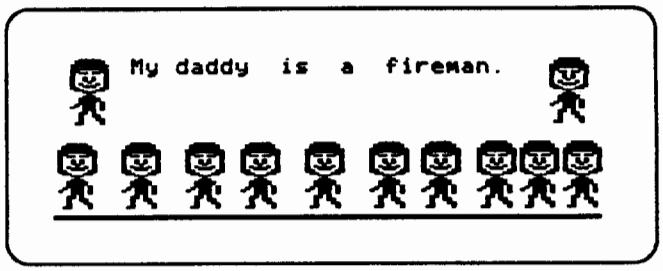
This work was supported in part by the National Science Foundation Grant TPE-8855541, the Education Development Center (EDC), and the Logo Action Research Collaborative (LARC), headed by Dan Watt and Molly Watt. The ideas and opinions expressed are those of the author and do not necessarily reflect the views of EDC or the National Science Foundation.

Donna Rosenberg was a participant in LARC 1989/1990, which was supported in part through a National Science Foundation grant. She was co-leader of the Boston group of LARC in 1990/1991.

Donna N. Rosenberg
Elementary Computer Specialist
Patrick J. Kennedy/Dante Alighieri/Curtis Guild Schools
East Boston, MA 02128







Goeffrey Brown, a first grader, wrote a "book" about his father and his father's work. He made his own shape for the fire.

An Electronic Community

by Glen L. Bull and Gina L. Bull

Much of the initial development of the programming language Logo took place in the 1970s. When microcomputers became available in the public schools in the late seventies and early eighties, few instructional tools were available. Much of the limited instructional software available consisted of computer-assisted instruction (CAI) written in BASIC. Logo filled the niche for a learner-based tool nicely, and was popularized by Seymour Papert's book *Mindstorms*, in 1980.

Evolution of a Logo Community

The success of the language led to the first international Logo conference, held in the early eighties. Tom Lough, founder of *Logo Exchange*, proposed that a conference be held. In the process of making preparations for such a conference, Tom received a call from Seymour Papert. Tom excitedly reported that the Logo development group at MIT had also been considering such a conference and extended Papert an invitation to come to Cambridge to discuss their joint efforts. Tom served as the cochair of the first Logo conference (along with Hal Abelson, if our memory serves us correctly), which became the first of several annual conferences held in Boston.

The series of Logo conferences in the mid-eighties was an exciting time for the Logo community. It was possible to meet and talk with the founders of Logo, and to meet other newcomers who had just discovered Logo. Serendipitous encounters abounded. For example, one evening we were talking with Howard Austin and were taken along to dinner with a group of Logophiles to learn about the relationship between Logo and juggling. (A discussion of this issue may be found on page 111 of *Mindstorms*.) Experiments with salt and pepper shakers endangered surrounding tables. (Later, Judi Harris was inspired to incorporate juggling as a major element in the first East Coast Logo conference, collecting *objets de juggling* months before the conference.) On another occasion in Cambridge, we were privileged to sit on the grass one sunny afternoon and listen to Dan and Molly Watt discuss their thoughts about Logo and learning cultures.

There are, of course, other attractions in Boston during the summer, which made the conferences appealing. Steve's Ice Cream, where spinning cans of ice cream churn in the windows, is always worth a visit in the summer. At one conference, a group of us took a picnic supper and attended a Boston Pops concert. Another year, a conference group attended a Boston Red Sox game and were privileged to watch Reggie Jackson knock a homer over the green monster. Even those group members who weren't sports fans enjoyed the soft pretzels, peanuts, and conversation that evening.

At each conference we seemed to meet new friends and discover new and unexpected Logo innovations. For example, at one conference we trailed along with a group as Mitch Resnick took us to a room filled with children playing with what later became LEGO-Logo. At another conference we followed Cynthia Solomon as she led us to the Atari Logo research laboratory, filled with wonderful ideas and concepts. For example, one version of Logo in this lab had buttons that could be moved around the screen and that could execute a Logo command or script. The first widely used commercial implementation of this concept was found in *HyperCard*, but the idea appeared in an experimental version of Logo before *HyperCard* was ever conceived.

The program and planning committees for the Logo conferences wrestled with concerns regarding ways to provide a conference that had some structure but still left room for chance encounters and Logo-like serendipity and discovery. One radical contingent suggested that a conference with no formal program or agenda might be most appropriate. A spontaneous Logo session that had coalesced in a hallway at the National Education Computing Conference (NECC) the previous year was offered as an appropriate model for a Logo conference. In the end, a printed program and agenda was devised, but Logophiles found their own means of devising spontaneous Logo discoveries and encounters not included on the formal program.

During this time the intrinsic geometry of the turtle was a pleasing and more intuitive improvement over the Cartesian coordinates required in BASICs of the time. Mice were not yet in widespread use, and the avalanche of paint programs triggered by *MacPaint* had not taken place. Word processors of the era did not provide a convenient way to integrate text and graphics, and the term "multimedia" implied use of a slide/tape presentation. As an increasing number of learner-oriented tools began to appear in the niche formerly held by Logo alone, users began to broaden the range of applications they used.

Developing a conference requires a lot of planning and support. A good conference requires a full year of development prior to the meeting, with numerousexpenses that must be underwritten prior to the conference. As members of the Logo community found their interests engaged by a variety of other activi-

ties, the need for a national conference devoted solely to Logo no longer seemed compelling, and the string of annual Logo conferences was ended, at least for a time.

A World-Wide Electronic Community

However, there is a way to meet and interact with other Logo users—electronically. Chance encounters with other Logophiles such as the ones we have described are now possible, and you do not even have to travel all the way to Boston. In last month's column, we mentioned that a vote was taking place to establish a Logo newsgroup on Usenet. That vote was positive, and the official name of the group is "comp.lang.logo".

Usenet discussion groups are widely distributed on the worldwide academic network Internet and can also be accessed through commercial services. In contrast to a bulletin board, which is usually found only at a single location, Usenet newsfeeds are distributed to thousands of sites around the world. Hence, even individuals in foreign countries who might not dial into a bulletin board in another country often participate in Usenet discussions.

If we dip into the Logo Usenet group at a random moment in time, we are likely to see discussions such as the following occurring. In this instance the newsgroup was accessed through Virginia's Public Education Network (PEN), a public school telecomputing network available without charge to any educator in Virginia. Depending on the system you use to access Usenet, details of the screen you see may differ somewhat. However, the underlying concepts will be the same for any system.

In sample screen shows at the bottom of this pace, case, discussions on a dozen different topics are listed. They include subjects such as "The Future of Logo," Logo for the Macintosh and for Microsoft *Windows*, LEGO-Logo, and a discussion with the intriguing title of "Logo has saved the day." Once an article is posted, others may choose to respond to that topic. For example, eight individuals have evidently responded to Topic 11, "Logo has saved the day."

Anatomy of a Usenet Article

To see how this might work, let's look at another posting in more detail. One parent has asked for suggestions about introducing his 10-year-old daughter to programming. The information in the heading of the posting tells us that he has posted this request for suggestions to the discussion group "misc.kids" as well as to "comp.lang.logo." One of the advantages of Internet is that it is possible to meet people from all different walks of life and with many different interests, and the cross-posting to "misc.kids" suggests an interest in comparing answers from "comp.lang.logo" with answers from individuals who may not be dyed-in-the-wool Logo fanatics.

```
Newsgroups: comp.lang.logo,misc.kids
From: thacher@unx.sas.com
Subject: Getting a 10 year old started
       in programming
Sender: news@unx.sas.com (Noter of
       Newsworthy Events)
Date: Thu, 6 Aug 1992 14:37:07 GMT

My ten year old daughter has been asking me to
teach her to program. She has had a little
exposure to computers at school, but there has
not been any real programming. I've gotten a copy
of mswLogo, but now I don't know what the next
step should be.

I would like her to be working independently as
much as possible, but it would be good to have an
idea of where to nudge her.

It would be great if there were an interactive
tutorial system that could teach the fundamen-
tals. One of the first systems that I learned on,
the Dartmouth Time Sharing System, had a "teach"
command. "Teach" would let anyone who wanted to
sit down at the terminal (a tty33) learn BASIC at
their own pace without anyone else having to help
them.
```

```
                        comp.lang.logo

      1  Future of Logo                          1 Larry Davidson
      2  MswLogo Version 1.2a is available         George Mills
      3  Getting MswLogo via (ftp) mail            George Mills
      4  Getting 10 year old started in program    Clarke Thacher
      5  Kid's introduction book?                  Nelson Bolyard
      6  Getting a 10 year old started           2 Rob Slade
->u   7  LogoMation - a logo-like language       3 Chuck Shavit
      8  MicroSoft Windows LOGO version 2.0 is     George Mills
      9  Logo for the Mac                          Allyn Weeks
   u 10  Object Logo                            2 Stephen C. Trier
   u 11  LOGO has saved the day :>              8 SDENARO@vm.poly.edu
   u 12  Lego logo...                           2 Cristobal Baray

      Select a topic, position the arrow then press the <Return Key>.
```

I'd like to hear from others who have experience with teaching their own children. I'd also be interested in some discussion about the merits of some of the other programming language choices: BASIC, SCHEME, SmallTalk, C, anything else? By the way, I'm using a pc clone running DOS 5 and windows.

One of the responses to this posting came from Rob Slade. The information in the heading of this response tells us that he is accessing "comp.lang.logo" through Computer Using Educators of British Columbia, Canada. Rob told us that he would not mind if we printed an edited version of his response in *Logo Exchange* to illustrate how newsgroups work. However, we enjoyed the anecdote that he included so much that we decided to print his response in its entirety.

From: rslade@cue.bc.ca (Rob Slade)
Subject: Re: Getting a 10 year old started in programming
Organization: Computer Using Educators of B.C., Canada
Date: Wed, 12 Aug 92 17:59:11 GMT

Give her the basic graphics commands: forward, back, right, left and clean.

When she asks (in a week or two), tell her about repeat.

When she asks, tell her about "To". (You'll likely have to cover the basics of the editor at that time.)

Answer questions: when you run out of answers show her the section of the manual that lists the primitives.

Don't worry about directing.

This may sound flip, but it is a considered opinion based upon observation and (in some ways sad) experience. I was able to participate in an experiment with Logo and grade three children at one time. Each "teacher" had at least one eight year old student, and could direct the study as desired. From reading "Mindstorms", it was obvious that Seymour considered that Logo could cure cancer, and I didn't quite go that far. I was, however, willing to put it to the test. Totally hands off, and let the child explore. By the second week, one of the other "teachers" had obtained a set of Logo worksheets. By the third week, all the other "teachers" had cracked: everyone was using the worksheets. I was climbing the walls: my little angel was doing nothing but filling the screen with masses of lines. But I was determined: stick to the concept, and if Logo (and Seymour) fell flat on its face, so be it. No skin off my nose. (But my student was so *stupid*!)

Sometime in the third week, in the space of two hours, my "stupid little airhead" surged ahead of the entire class. By the fourth week, she was correcting and improving the procedures of the rest of the class. By the end of eight weeks she had thoroughly mastered procedures and modular programming, was dealing with recursion and list processing. She had even discovered, on her own, the "Total Turtle Trip Theorem", which I had frankly considered to be impossible. (I was very careful about *not* giving her any info on it. She didn't know anything about "degrees", but she did discover that a closed figure required a total of 360 in "turns". Blew me away, the day she did a perfect pentagon ... on her very first attempt.)

Let her explore. Let her use Logo as long as she wants. It's got everything she'll need, conceptually. The language she might use in eight years may not be invented yet. Don't sweat it.

Vancouver ROBERTS@decus.ca
Institute for Robert_Slade@sfu.ca
Research into rslade@cue.bc.ca
User pl@CyberStore.ca
Security Canada V7K 2G6

"A ship in a harbour is safe, but that is not what ships are built for." -John Parks

There are several significant points, we think. One is that a parent who might not otherwise come into contact with anyone from the Logo community can post a question and receive a personalized response to a question about getting started with Logo. Internet is the world's computer network, accessed by more than 25 million users. The opportunity for interactions between long-time Logo users and others who may know of Logo only through turtle graphics, if at all, is high. Of course, with more than 1,000 different discussion groups, only those who are interested in educational computing are likely to read the postings for the Logo discussion group.

However, the most attractive aspect of this electronic Logo conference for us is that it potentially provides the opportunity for chance encounters and discussions in a spirit similar to that found at the original Logo conferences in Cambridge. If you pop into the conference at any given time, you might encounter Brian Harvey, or Ken Johnson, or a professor from Oxford University or the University of Kaiserslautern in Germany, or a researcher from the Human Interface Laboratory in Washington state or from the Los Alamos National Laboratory or from companies such as AT&T, Digital Equipment Corporation, or Silicon Graphics. These were the sites from a random sample of some of the recent postings as we skimmed through the listings for the Logo discussion group.

Many of these contributors are well known in the Logo community. Brian Harvey, for example, is well known for his series of books *Computer Science: Logo Style.* If you have a question about the implications of a programming technique in Logo, there is possibly no one better qualified to answer the question. Ken Johnson has run a Logo mailing list on Internet for many years from the Artificial Intelligence Applications Institute in Scotland. (His motto is the disarming "Bugs-R-Us.") However, an equal number are simply parents who want information about using Logo with their children. The new Logo newsgroup makes it easy to discuss Logo issues with both experts and novices.

There is also the opportunity for serendipitous encounters. As we began talking with Rob Slade, we discovered that he was one of the developers responsible for the World Logo Conference in 1985. The World Logo Conference was notable because it combined an on-site conference with an electronic component that stretched around the world. Although the conference was held in Canada, Logo contingents around the world participated. At prescheduled times, Logo presenters such as Seymour Papert came on-line to talk electronically with the audience. As we participated in the conference, one of the most fascinating aspects was the process of watching participants from around the world come on-line for the first time as the earth turned and the sun rose in their region. To our knowledge, this was the first conference of this kind.

Logo-Like Qualities of Usenet

It seems appropriate to have a Logo discussion group on Usenet because Usenet has many Logo-like qualities. Usenet is a cooperative group of computer users who exchange information organized around "newsgroups." The term "newsgroup" is slightly misleading because newsgroups are actually electronic discussion groups. Usenet was founded by two graduate students at Duke University and a third at the University of North Carolina. It is difficult to say how many sites worldwide participate in Usenet, since many networks beyond Internet carry Usenet newsgroups, and not all sites on Internet use Usenet feeds (though most do). However, Internet currently has approximately 600,000 servers, so it is probable that Usenet discussions are carried on more than a half million systems. (In contrast, CompuServe has only a single server, located in Columbus, Ohio.)

There is no central organization governing Usenet. Instead, each site independently decides which newsgroups to carry. All these features of Usenet give it a surprisingly Logo-like feel. The Logo culture has tended to favor grass-roots initiatives that are perhaps the antithesis of the top-down structure of Computer-Assisted Instruction (CAI). Usenet represents one of the most successful examples of a grass-roots culture. Usenet is governed by informal rules about interacting courteously with others over the network (sometimes called "Netiquette"), but no central administrative agency exists. Chapter 4 of *Zen and the Art of the Internet* (Kehoe, in press) provides a good overview of Usenet newsgroups for those who are interested in further information on this topic.

Access to "comp.lang.logo"

At this point you may be asking yourself how one gains access to this electronic community. You will, of course, need access to a computer and a modem. If you are a teacher, several states provide access to Internet, which may include access to Usenet discussion groups in some instances. The Virginia Public Education Network (PEN), the Texas Education Network (TENET), the Florida Information Resource Network (FIRN), NYSERNET in New York, the Big Sky Telegraph and WESTNet in the West, and the California public school network all provide teachers with access to Internet. In Cleveland, any citizen can obtain an account on the Cleveland Free-Net at no charge, and a number of other communities have adopted community networks modeled on the Free-Net system.

Most (though not all) universities in the United States provide free Internet access to faculty and students. If you are at an educational institution, ask your computer center whether Usenet newsgroups are available. If you are a teacher, you may want to ask a nearby university if you may have an account. Some universities will provide public school teachers with Internet access as a public service. (In fact, Virginia's Public Education Network began in this manner.) Other universities charge a small fee to cover administrative costs. If you are the first teacher to inquire about this possibility at a university, you may find it helpful to take a faculty member from the school of education with you to support your request when you ask.

There are also commercial services that provide access to Usenet newsgroups. We have not used these services but have noted that a number of contributors to Usenet newsgroups evidently gain access to Internet through subscription to commercial services. If you are using a commercial service to read Usenet newsgroups such as comp.lang.logo, we would like to hear from you about your level of satisfaction with the service, what the costs are, and so forth. If any *LX* readers are using such services, we will summarize your conclusions in a later issue.

In the meantime, we are pleased to report that another place for the Logo community to meet and interact has been established and appears to have had a healthy beginning. From time to time in future columns we will report on highlights from comp.lang.logo and report its progress as this electronic community grows and evolves.

Reference

Kehoe, B.P. (in press). *Zen and the art of the Internet.* Englewood Cliffs, NJ: Prentice Hall.

Internet Addresses:
GBull@Virginia.edu, Gina@Virginia.edu
BITNET Addresses: GBull@Virginia, Gina@Virginia

MSWLogo Available

MswLogo Version 2.1 for MicroSoft *Windows* 3.X is Ready. (9/15/92)

Location: itrc::/millspub/mswlogo21.zip (this is a DEC internal location) or cher.media.mit.edu on pub/comp.lang.logo/programs/mswlogo or

Send the following message to: ftpmail@decwrl.dec.com

```
connect 18.85.0.47
binary
uuencode
chdir pub/comp.lang.logo/programs/
    mswlogo
get mswlogo12a.zip
quit
```

New features/bugs fixed:

- Cleaned up some errors in help
- New Logo commands
  Getpenred,getpengreen, getpenblue, penred,pengreen,penblue
  Getscreenred,getscreengreen,...
  Getfloodred,getfloodgreen,...
  Scrollx, scrolly (Logo procedures can now control the scrollers)
  Bitcut/bitpaste (very powerful in logo).
  Infite range zoom.
- Jagged turtle fixed.
- Cleaned up garbage and order in error messages.
- Slimmer commander.
- Save and restore image now works again and it uses ".BMP" format.
  This opens up a lot of possibilties. You can load in work from other applications and visa-versa. Instead of posting your childs work on the frige you can make it you windows background !!!.
- Printing fixed on most printers.
- And as usual lots of little bug fixes.

To Install it:

1) UnZip MSWLogo21.ZIP into a directory (e.g. c:\logo)
2) UnZip LOGOLIB.ZIP (included in MSWLOGO21.ZIP) into LIB (e.g. c:\logo\lib)
   Note: If you do not use c:\logo as the root you must set the environment variable LogoPATH to where you put it (e.g. set logopath=d:\games\logo in autoexec.bat)
3) Copy BWCC.DLL (included in the .ZIP) to anywhere in your PATH. The could be where you place Logo (as long as it's in your PATH). When I say PATH here I don't mean LOGOPATH. The desired place is in your root windows dorectory (e.g. c:\windows).
4) Start Windows and add Icon as follows:
   a) Select the applications (or other) icon box.
   b) Go into Program manger FILE menu and click on New.
   c) Select Program Item and click OK.
   d) Enter the command for logo (e.g. c:\logo\logo).
   e) Include a working directory (e.g. c:\logo\work) or you'll have ".lg" files all over the place (this applies to 3.1 only). I'll try to come up with a better solution soon.

Start it from Windows by double clicking the Logo Icon

Start it from DOS by typing "win logo" at DOS prompt (assuming logo.exe is in your PATH).

Try typing "DEMO" in the "Input Box" (out of bounds at end of demo is normal). Note: "DEMO" is a logo program in the library, it's not an internal command.

Standard features:

- Supports Text
- Cut and Paste.
- Floodfill.
- Penwidth.
- Save and restore images in .BMP format.
- Screencolor (background).
- 16.7 million pen and background colors (using windows dithering).
- Standard Windows Hypertext Help.
- Standard Windows Printing.
- Supports separate library and work area.

Bugs / Basic Missing features:

- Real mode is not supported.
- Pause and Trace buttons are there but not implemented yet.
- Pots, goodbye still not working.
- Printing on (very few) printers still doesn't work.
- Need to make save/restore image smarter (less memory, disk and cpu).
- Logo does not use the capabilities of 256-color Video Drivers.
  NOTE: If your using a Video driver that supports more than 16 colors
  and find Logo sluggish try reducing to a 16-color.
  If your disk is thrashing while running Logo you probably do not have enough physical memory to support the number of colors your driver is using.
  I'll try to make Logo more flexible around this (e.g. variable size work surface).

Wish list:

- Need sound/multimedia support.
- Needs debug support.
- Undo/Redo.

The Core of this Version of Logo comes from a project done by Boston Childrens Museum and Lincoln Sudbury Regonal High School. The Port to Windows was done by George Mills Digital Equipment Corporation. Digital Equipment Corporation takes no responsibility for the software.

There is also a DOS version (with out the added functionality added to MswLogo) is available. Same location as MswLogo.

Exploring the CHAR Primitive of *LogoWriter* on Your IBM

by Charles E. Crume

Recently, as I was working on a *LogoWriter* program, I noticed that when I tried to print a number followed by a parenthesis, *LogoWriter* displayed a space before the parenthesis. After some thought, I realized that parentheses are special characters. After consulting my *LogoWriter* manual, I found a primitive named CHAR that accepts a decimal value between 1 and 255 and returns the character corresponding to that value. The manual listed the right parenthesis as having a decimal value of 40. When I executed the command:

```
PRINT (WORD 1" CHAR 40 CHAR 32 "CIRCLE)
```

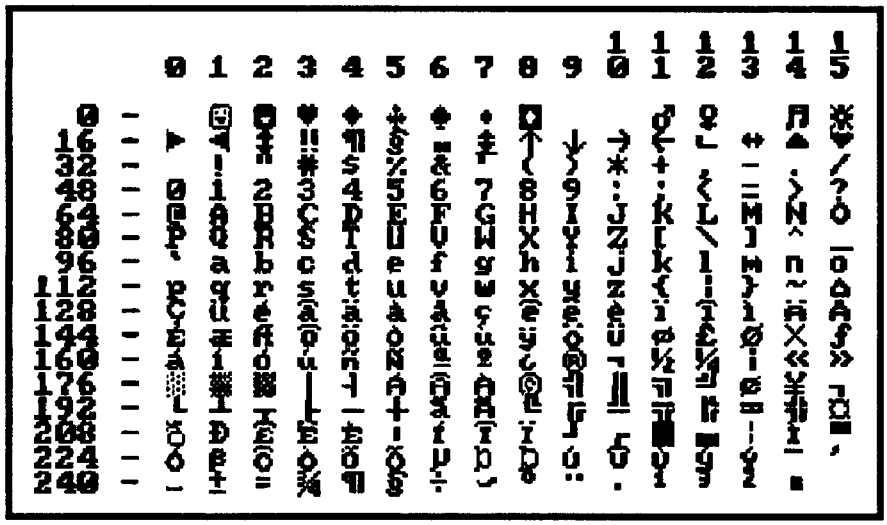
LogoWriter displayed:

```
1) CIRCLE
```

I could display a parenthesis adjacent to whatever I wanted. Now, however, I was interested in what else could be done with the CHAR primitive. To simplify the task, I wrote a procedure to display all 256 characters on my IBM PC. The procedure is called CHARSET, and the source code is shown below:

```
TO CHARSET
RG
HT
CT
REPEAT 27 [INSERT CHAR 32]
PRINT [1 1 1 1 1 1]
REPEAT 7 [INSERT CHAR 32]
PRINT [0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
   5]
PRINT [ ]
ROW 0
TOP
END

TO ROW :VALUE
IF :VALUE = 256 [STOP]
INSERT CHAR 32
IF :VALUE < 100 [INSERT CHAR 32]
IF :VALUE < 10 [INSERT CHAR 32]
(INSERT :VALUE WORD "- CHAR 32)
ROW COLUMN :VALUE :VALUE + 16
END

TO COLUMN :NEXT :LAST
IF :NEXT = :LAST [PRINT [ ] OUTPUT
   :LAST]
IFELSE (OR (:NEXT = 9) (:NEXT = 10)
   (:NEXT = 13)) [(INSERT CHAR 32
   ")] [(INSERT CHAR :NEXT ")]
COLUMN :NEXT + 1 :LAST
END
```

The characters are arranged in a table consisting of 16 rows and 16 columns. Each row and column is numbered to facilitate determining the value of any particular character. The values down the left side (0 through 240) are the beginning value for characters in



that row. The values across the top (0 through 15) are the offset within a row. A character's value is computed by adding the row value and the column value.

For example, the digit 3 (4th row, 4th character from the left) has the value 51 (row value 48 plus column value 3). The copyright symbol (12th row, 9th character from the left) has the value 184 (row value 176 plus column value 8).

CHARSET displays a graphic symbol (i.e., character) for every value except 0, 9, 10, 13, 32, and 255. A blank space is shown for these values because:

- The values 0 and 255 have no character representation
- The values 9, 10, and 13 cause the cursor to perform a horizontal tab, line feed, and carriage return, respectively—actions that would affect the table
- The value 32 is truly a space.

Examining the table shows a number of interesting and useful characters. Among them are:

- The four suits in a deck of cards—hearts, diamonds, clubs, and spades (values 3 through 6)
- The biological symbols for male and female organisms and cells (values 11 and 12)
- Four directional arrows (values 24 through 27)
- A copyright and registered trademark symbol (values 184 and 169)
- A plus/minus sign for indicating probability error (value 241)
- Numerous international characters.

Some sample *LogoWriter* commands making use of these symbols are:

```
(PRINT [RESULTS ACCURATE TO WITHIN]
   CHAR 241 [3%])
```

displays

```
RESULTS ACCURATE TO WITHIN ± 3%
```

The command:

```
(PRINT WORD "4 CHAR 4 WORD "9 CHAR 6
   WORD "J CHAR 4 WORD "3 CHAR 5
   WORD "Q CHAR 3)
```

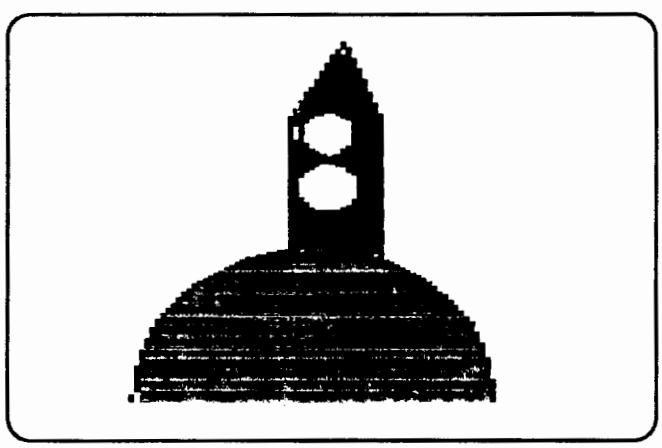
displays

```
4♦ 9♠ J♦ 3♣ Q♥
```

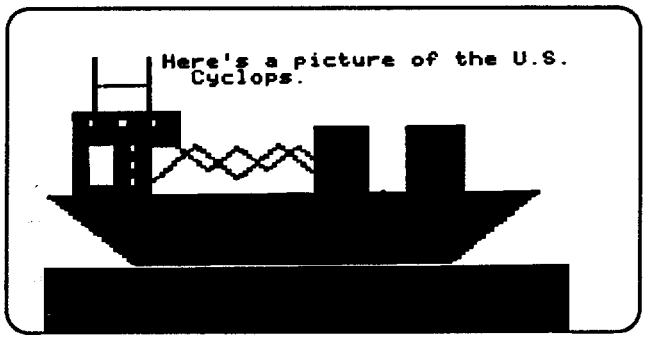
Some of you are probably thinking, "But I could create a shape or draw these characters using turtle graphics." After all, mixing text and graphics is one of *LogoWriter*'s major features. Technically speaking, one could use shapes or turtle graphics. However, this could be quite tedious and messy. Imagine the instructions needed to move the turtle to each place in the text where a parenthesis was to go and stamping the shape at that location and the work involved if the format of the text changed—all turtle movement instructions would require changing. Using CHARSET characters facilitates using international characters for foreign language programs or for writing a card game.

Charles E. Crume is a software development programmer at a medical research laboratory in Cincinnati, Ohio. His continuing interest in Logo and educational computing stimulates the writing of articles and utility programs.

Charles E. Crume
810 Matson Place, Apt. 504
Cincinnati, OH 45204



Raymond Longobardi, a third grader, did research on Mars. This page from his hypermedia presentation shows how one day we may be able to land on Mars and colonize it.



Jonathan Mummolo, a third grader, did a hypermedia presentation on the Bermuda Triangle. The Cyclops is one of the ships that disappeared in the Bermuda Triangle.

African Textiles or The Weaving Turtle

by Orlando Mihich

Introduction

African textiles, as African masks, figures, pottery, and body ornaments and painting are all expressions of the rich African cultural heritage. These creations of art, through structural relationships of form, design, and colors express feelings that cannot be voiced any other way. Picasso, influenced by African art, was capable of breaking the western figure and starting a revolution in western art. He can be considered the first western genius coming out of Africa. Modigliani, and Giacometti, created work deeply influenced by the art of Africa. The sculptures of the Dogon artist and Giacometti are strong creations which speak the same universal language.

Textile

The word textile, comes form the Latin "texere," to weave. African textile is rich in colors, and is endlessly creative in geometrical design. Hand made textile, from the hands of weavers, spinners and dyers, is related to ancient pre-colonial traditions. Particular colors, shapes, and decorations may have political and ritual significance. Textiles are used not only to make articles of clothing, but also for decoration, to dress a house or a shrine during important events, or simply as gifts. The meaning of color varies from place to place. In Benin, the color red is part of the ceremonial court dress, among the Ebira the red color is associated with success, while in Madagascar red is applied to burial cloths. In most of West Africa, Ethiopia, East Africa and Zaire, all weaving is done by men. In North Africa and Madagascar, all weaving is done by women. In other areas, both men and women weave. The most common materials used in weaving are wool, silk, cotton, bast, and raphia. Indigo is the most common dye, and is obtained from various plants of the genus Indigofera. Indigo gives various shades from pale blue to deep black. Other colors like red, yellow, blue, green, brown, and black are available from local vegetable and mineral sources.

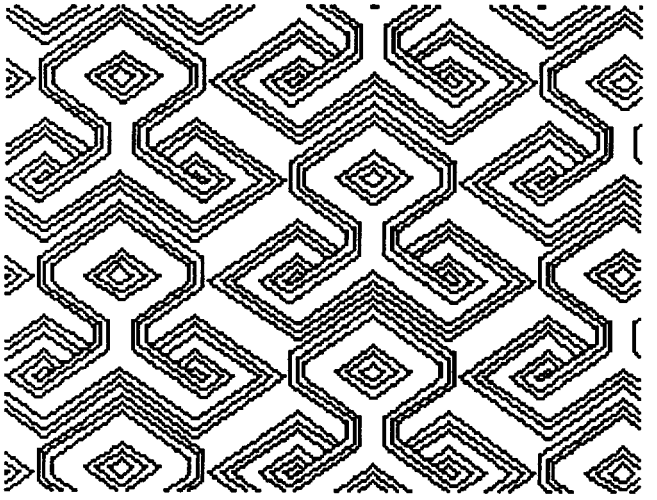
The Logo Textile

Islamic tessellations, Maya glyphs and weaving, African masks and textiles make up an art that speaks directly to the Logo programmer. It is impossible to visit a Mayan site without mentally creating at least a simple procedure and repeating the design. The same

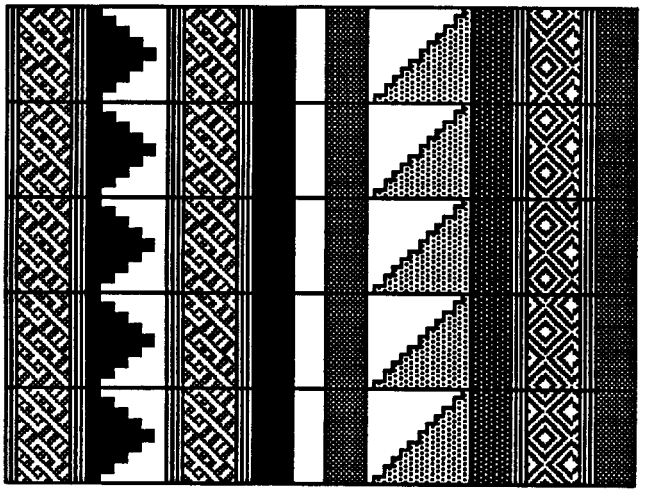
happened to me and my students with African textiles. My students taught the turtle to "weave" and generated several screens. They did not simply copy the designs, but used their own color schemes, "stitched" the various parts of the design in their own way, and changed backgrounds to see the effect of different "materials."

The following screens are reflections not only of artistic and aesthetic feelings but also of knowledge of mathematics and geometry.

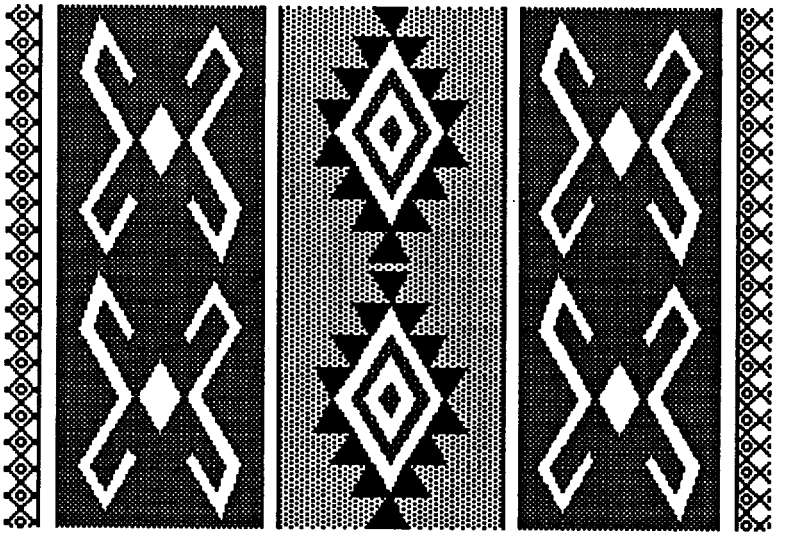
Orlando Mihich
Science and Computer Teacher
339 Pacific Avenue
Jersey City, N.J. 07304



Embroidered raphia textile from Kuba, Zaire.
Myckele Spencer



Cloth used in men's ritual celebrations.
Woven by the Mende people of Sierra Leone.
Michael Haistock



Woolen textile from Niger.
Michael Toribio and Coedell Page

1993 International Logo Conference

A Working Conference for Educators

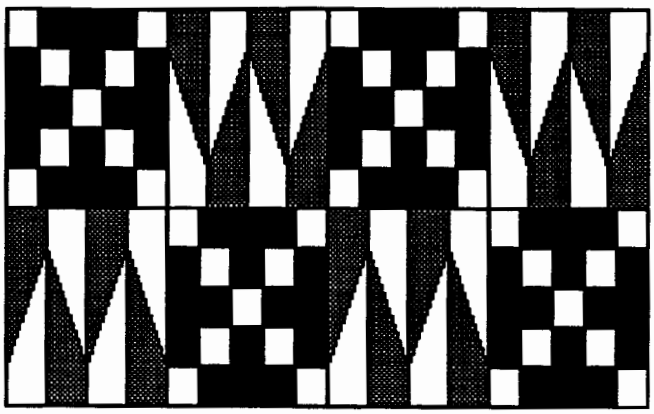
Methodist Ladies' College
Melbourne, Australia
July 4-7, 1993

Keynote Speakers: Idit Harel and Barry Newell
Invited Speakers: Dan Watt, Molly Watt, Brian Harvey, E. Paul Goldenberg, Linda Polin, Leslie Thyberg, Gary Stager

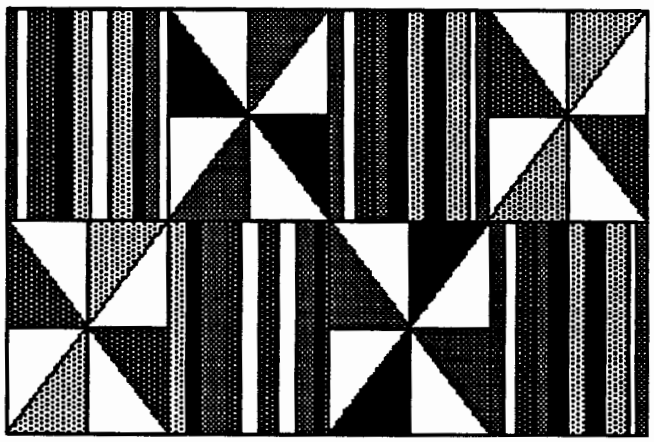
Interactive Presentations
Collaborative Workshops
Free Warmup *LogoWriter* Preconference Workshop
For more registration or presenter information, contact:

1993 International Logo Conference
MLC Community Education
207 Barkers Road
Kew, Victoria 3101 Australia

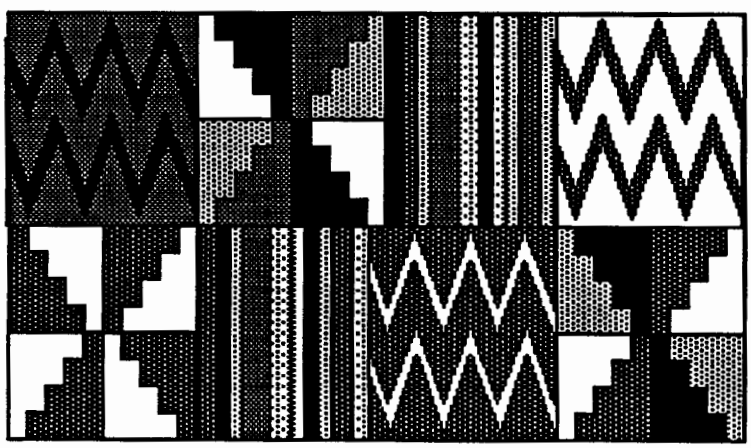
Voice: 03 810 1412
Fax: 03 819 2345
Email: K0331@Applelink.apple.com



Cloth woven by the Asante people of Ghana.
Michael Hailstock



Ceremonial Kente cloth from Ghana.
Myckele Spencer



Kente cloth woven by the Asante people of Ghana.
Michael Haistock

Will Logo Survive?

by A. J. (Sandy) Dawson

How many times has the question "Will Logo survive?" been asked? And if it has been asked so often, why explore it again? The answer to the latter question is because this time the former question was being asked by Alan MacLeod, an elementary school teacher who dismissed Logo when it first appeared but who because of recent experiences in a graduate course looked again at Logo, and the debate that surrounds it.

I met Alan MacLeod on-line. He was registered for a course I was teaching to a cohort of graduate students—a group that gathered infrequently in Kelowna, British Columbia, some 450 kilometers from Vancouver, where I live and work. The course was to be offered electronically, and I was to have just three face-to-face meetings with the group. Alan was intrigued by comments about Logo—which led him to "dust Logo off" and to take another look at it. This experience eventually enticed him to look more rigorously at the "why" behind the debate about Logo. His story is told below.

From Turtle to Mouse...Will Logo Survive?

by Alan H. MacLeod

The Manual in the Cupboard

I have been teaching at the elementary level for 10 years. Prior to this I taught math and physics at the secondary level. I had never heard of Logo. It was not until after four years of teaching at the elementary level that I discovered, on the top shelf of a cupboard in the computer lab, a dust-covered, hard-bound manual with the inscription *Terrapin Logo*. The manual also consisted of 14 disks, each in its own pocket. A quick browse through the manual brought only a mild sense of curiosity; the manual appeared to deal with some form of elementary computer programming language that was used to move an object called a turtle, that didn't look like any turtle I had ever seen. It appeared quite challenging for the elementary-level students, and I thought that my students would not be able to handle it. I put the manual back on the shelf where it remained, most likely to remain untouched for another five years.

Introduction to the Problem

It was not until just recently while I was taking a graduate course in education that I began to see references to Logo in my readings. I was surprised to learn from these readings that Logo programs were being used at the primary and intermediate levels. I was curious as to the controversy that surrounded Logo. I retrieved the manual from the cabinet, once again wiped the dust from its cover, and read the first few pages.

The programming steps did not appear difficult: DRAW, FD 100, RT 90 were commands that would make the turtle move—something fourth-grade students could handle easily enough, I thought. I made the decision to introduce Logo to some of the more capable students in my class. I thought they would, with practice, become peer tutors for others in the class. The students appeared eager when I told them that we were going to explore Logo together. The first lesson went well. All students were on task, as I expected, drawing shapes of all kinds. The more adventurous dared to program turtle steps in the thousands and were delighted with the results. I was pleased with the first lesson. It was a good one.

The next lesson did not go as well. I deviated from the manual and had them draw squares and rectangles. They were to think about the rules they used to draw these shapes. My next request of the students involved one of those "while you're at it why don't you try..." kind of teaching methods. I often reflect on that moment because it placed doubts in the minds of both myself as a teacher (a good one I thought), and the students as to their capabilities. After that lesson, I became increasingly aware of some students' growing resistance to Logo lessons. There was a problem somewhere. I was not sure whether it was me or Logo. I was compelled to delve into the beginnings of Logo and read as much about it as I possibly could. Logo, I found, has received much praise and criticism. After much reading and much reflection, I sense that I now know where the problem lies. It lies not in Logo itself: it lies in how educators interpret Papert and in the misconceptions researchers hold about what counts as learning.

Getting to the Root of the Problem

When I first discovered the Logo manual, it appeared to me that Logo was just another computer language, but in a more simplified form. Papert (1980), however, emphasized that Logo was more than that. He saw Logo in a Piagetian sense: Logo was to be used as an "object-to-think-with." Papert, nonetheless, cautions the reader that although Piagetian learning to him

(Papert) means "learning without curriculum," it does not mean "free-form classrooms or simply leaving the child alone." Papert's statement seems to imply that there must be some intervention or guidance in a student's interaction with Logo. The question, however, is guidance by whom, and how much intervention might there be? His statement is open to interpretation. One interpretation is made by Sheingold (1987): "According to Papert, Logo is an environment in which children can learn problem-solving methods without the intervention of teachers." Indeed, one would assume that since Papert takes his inspiration from Piaget, Logo must be something convivial, similar to Bigum's (1987) convivial spreadsheet—an object that both teacher and learner can enter and explore together in a kind of educator-educatee partnership. Emihovich (1990) contends that Papert did not emphasize enough the importance of teacher intervention, and that "although he recognized that Logo must be seen as a cultural process, his precise meaning is not clear. In contrast, it is clear that he saw the teacher as having a peripheral role, if any at all." How much or how little guidance is needed to learn Logo would depend on the interpretation of what Paper means by "not leaving the child alone."

The degree of success teachers have had with Logo in their classrooms may very well depend on how the teachers were sold on using Logo in the first place. To most children a turtle is a harmless, unobtrusive creature that can be easily picked up, turned around, and made to go in another direction. The name *Terrapin Logo* leads one to assume that Logo is friendly and can be managed alone without any difficulty by all children.

Certain rules of Logo programming must be learned prior to tinkering in Papert's Logo micro-world. To assume that even with teacher intervention all children will acquire these rules sufficiently to begin (and to enjoy) their explorations in a Logo mathworld may lead to disappointment for both teacher and learner, and eventually lead to rejection of Logo as having little if any practical use. Turkle (1984) makes the observation that "Logo has fallen into disuse in many schools since the excitement over its unlimited potential has eroded."

The question remains as to what led teachers to assume that Logo was exploratory in nature, with perhaps only minimal intervention. Was it because of how they interpreted Papert? Or was it because of how others interpreted Papert for them?

Part of the answer may be found in looking at how Solomon (1986) interprets Papert:

> Often Piagetian learning à la Papert is interpreted to mean that children do not need help from experts, that exploring Logo without human intervention of an expert is sufficient. This was not Papert's intention. Logo provides an environment and culture in which novice and expert can find common ground to discuss their research and the bugs they encounter. (p.131)

Although it appears that Solomon (1986) may have correctly interpreted what Papert probably meant by not leaving the child "alone," she does recognize that teachers are having difficulties with Logo in the classroom. As a remedy to this problem, Solomon suggests that there be centers for teachers who want to integrate computers into their children's learning environments. It would be at these centers that teachers would fully realize the possibilities that Papert had intended for Logo:

> The center will be a place to explore and develop personal styles of learning. It will demand a deep personal commitment from each of the participating teachers. This learning center will draw on the ideas of Papert and Kay in developing computer environments and also on the work of Suppes, Davis, and Dwyer.

It is fair to say that Solomon's proposal of learning centers would go far in preparing teachers to use computers in learning environments. Teachers would have the opportunity of exploring not just Logo learning environments, but any other computer software designed for educational purposes.

Papert did not make any guarantees that Logo, in terms of what he envisioned it was to achieve, would prove to be a success. According to Papert (1980), certain conditions would have to be met prior to implementing Logo in any classroom:

> First, that all children will, under the *right* (italics mine) conditions acquire a proficiency with programming that will make it one of their more intellectual accomplishments. Second, that the right conditions are very different from the kind of access to computers that is now becoming established as the norm in schools.

Papert's failure to identify just what were to be the "right conditions" for Logo to work, and his failure to clarify what he means by "not leaving the child alone," leaves much open for interpretation. It would seem safe to assume that many teachers have incorporated Logo into traditional classroom settings of 30 or more students, and had only a token number of computers to accommodate these students. If classes were held in computer labs, computer time was limited to one or two 1-hour classes a week. Such settings might be contrary to what Papert would have considered the "right" conditions, yet it was at these kinds of sites where Logo was researched! It is probable that inconclusive and invalid results as a result of research done on Logo in settings that Papert would not have considered to be the "right conditions" may have done much to harm Logo.

Clements (1990) makes some headway in addressing the problems associated with Logo. He contends that neither the "exposure approach nor the conceptualized framework approach" is satisfactory. Instead, he suggests using the mediated conceptual framework approach. Clements' notion of mediated conceptual framework clearly emphasizes the importance of the guiding role of the teacher in using Logo. According to Clements, one of the positive effects of Logo is that of helping students to learn how to find and correct their mistakes (debugging). Significantly, he adds that "these and other studies with positive results employed much teacher mediation." Other positive benefits of Logo, remarks Clements, include facilitating social interaction and the focusing of that interaction on learning, criticizing in a helpful way and appreciating the work of others, and enhancing self-concepts. Ultimately it has been established that the teacher must not only be familiar with the Logo language, but he/she must be prepared to mediate extensively.

Emihovich (1990) reports Papert's reaction to studies done on Logo that had to do with "effects" on students and their learning. She remarks that "Papert derided the use of questions such as 'what is *the* effect of *the* computer on cognitive development?' and 'Does Logo work?'" According to Emihovich, Papert referred to such questions as "technocentric thinking." She also notes that "the context for human development is always a culture, never an isolated technology."

Emihovich's report confuses the question even more. She contends that neither the researchers nor Papert had "the appropriate conceptual starting point" to begin the study of computer use in schools. On Papert's point of view, Emihovich remarks:

> Although Papert was on the right track in emphasizing the cultural aspects of computer use, his frame work was too diffuse; it is almost impossible to move from his sweeping statements down to the level of the classroom use, and then to locate classroom computer use within a wider social context.

Looking to Emihovich for a solution as to how Logo could have been better researched has only led to more questions.

Empirical methods, beyond showing improved scores on standardized math scores (Clements , 1990), have failed to show that Logo facilitates the cognitive process in any significant way. Single-case, ethnographic, studies such as those done by Lawyer (1986) on Robby and Turkle (1984) on her Austen students, appear to show promise for Logo. In such studies, process rather than product is emphasized. According to Emihovich, Papert preferred anecdotal, ethnographic methods of research. To Papert, Logo was meant to be an object-to-think-with. Logo was to be a microworld into which a student could enter and grasp mathematical concepts. There was no guarantee by Papert that using Logo would boost test scores. Nor was it his intention that Logo would have any effect on planning skills.

A Second Chance for the Turtle?

Research into the effectiveness of Logo and other Logo-type computer environments require a new approach to learning. According to Wellburn (1991), this new approach is based on what cognitive science has provided in understanding "the interaction of an individual's mental functioning and the complex contexts in which it occurs." Wellburn further states that from the cognitive perspective it is "active learning" that leads to cognitive change:

> Active learning (including strategy use), the transferability or generalization of learning, and the importance of the context for learning are interrelated areas that have received much attention in cognitive research... An adept learner makes use of cognitive strategies for learning and use of metacognitive strategies to monitor his/her progress.

Perhaps Logo facilitates active learning. According to Reid and Green (1989), active learning consists of five phases: engagement, exploration, transformation, reflection, and presentation. They further suggest that assessing learning in any of these five stages requires anecdotal note-taking by teachers of learners involved in various learning tasks, interviewing students, and having students make journal entries. It is probably in the context of active learning that the effects of Logo should be researched. There are still questions that remain unanswered.

Certainly, the mouse technology is here to stay for a time, and with it has come interactive software that is much more readily accessible than Logo. But it is also common knowledge that turtles are among the longest surviving group of creatures on earth, and hence the Logo turtle's demise should not be trumpeted just yet!

References

Clements, D. H. (1990). Logo: Search and research. *Logo Exchange, 9*(2), 32-35.

Emihovich, C. (1990). Technocentrism revisited: Computer literacy as cultural capital. *Theory Into Practice, 24*(4), 226-234.

Lawler, R. W. (1986). Natural learning: People, computers and everyday number knowledge. In R. W. Lawler et al., (Eds.), *Cognition and computers: Studies in learning*, (pp. 9-88). New York: John Wiley and Sons.

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

Reid, J., & Green, B. (1989). A curriculum framework: Teaching for powerful learning. *Australian Journal of*

Reading, 12(3), 179-209.
Solomon, C. (1986). *Computer environments for children.* Cambridge: The MIT Press.
Turkle, S. (1984). *The second self: Computers and the human spirit.* New York: Simon and Schuster.
Wellburn, E. (1991). *Information, telecommunications and learning: A Review of the literature.* Unpublished manuscript.

Alan MacLeod is a graduate student at Simon Fraser University. He has been part of a cohort of graduate students completing their studies in conjunction with the Southern Interior Telecommunication Project.

Email: Alan_MacLeod@sfu.ca

Postscript: Sixth Logo Mathematics Education Conference

This column went to press at the same time that the Sixth Logo Mathematics Education (LME6) Conference was drawing to a close in Vancouver. There, the question of the demise of Logo was not a central concern. Instead, the question was, what is next? Andy DiSessa's work on Boxer was certainly one possible answer, as was Uri Wilensky's *Logo, but there were other answers as well. Moreover, the question and answers were broadened to include the issue of what Logo-like software environments might be like, and here such things as Isetl, Maple, Cabri, and the Geometer's Sketchpad were explored and debated. It seems that the spirit of Logo is alive and well, but that is a story for the next column!

Sandy Dawson is an associate professor of mathematics education at Simon Fraser University and is director of that institution's teacher education program. His most recent research interests center on the areas of LEGO/Logo and the exploration of what mathematics lessons with a constructivist or humanistic focus might look like.

A. J. (Sandy) Dawson
Faculty of Education
Simon Fraser University
Vancouver, BC.
Canada V5A 1S6
Email : Sandy_Dawson@sfu.ca

More Than 100 Colors Possible in *LogoWriter*

by George Trombley

If you have ever used *IBM Logo* and then started using *LogoWriter*, I am sure you appreciated the 15 colors available to you in *LogoWriter* as opposed to the 4 available in *IBM Logo*. I also was impressed. However, soon I was craving more. Through experimentation, I discovered a simple way to create up to 225 color combinations.

With the primitive SHADE, this is possible. Carefully follow these steps:

1. Flip to the SHAPES screen and make a checkerboard shape, coloring every other square.
2. Leave the SHAPES page, remembering the number of the shape you used to make your checkerboard.
3. Clear the graphics screen and start with a blank screen.
4. Set the pencolor to a desired color. We will call this color C1.
5. Make sure the pen is down and type

```
FILL
```

6. Set the turtle to the checkerboard shape.
7. Set the pencolor to the color you wish to blend with C1. This color we will refer to as C2.
8. Now for the blending. Type

```
SHADE
```

What you have done is to blend C1 with C2 to make C3. Amazing! Right? Well, maybe not so amazing, but it will surely increase your graphics capability.

There is more to this than there appears to be. Using this blending technique, you can create "textured" objects. First, enter this procedure:

```
TO CUSTOM.FILL :FIRST.COLOR
   :SECOND.COLOR :SHAPE.VALUE
HT
SETCOLOR :FIRST.COLOR
PENDOWN
FILL
SETSHAPE :SHAPE.VALUE
SETCOLOR :SECOND.COLOR
PENDOWN
SHADE
END
```

You can use CUSTOM.FILL to blend your colors by simply making SHAPE.VALUE the checkerboard shape's number. If you create different textures on the SHAPES page and you use those shapes for SHAPE.VALUE, you can texture your art using the CUSTOM.FILL procedure.

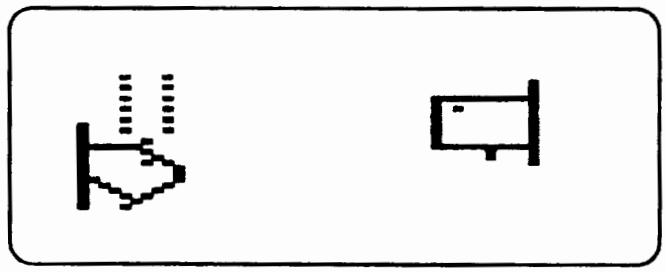
For example:

```
CUSTOM.FILL 12 15 1
```

(Checkerboard Shape is 1) By using the Checkerboard shape for SHAPES.VALUE, you are able to use CUSTOM.FILL as a color blender.

Any other basically abstract shape would either texture or "wallpaper" your art. Experiment. But, remember, CUSTOM.FILL doesn't work like the normal fill. You can't just simply refill an already custom-filled area—first you need to use Reset Graphics (RG) Have fun!

George Trombley
R.E. Manning
Edgren High School
APO 96519



Serens Shulman, a second grader, did some research about flounder. Then she made an animation showing the flounder (with bubbles) being attacked by a larger fish. When the large fish gets close, the flounder changes to the same color as the background and disappears.

Squares and Rectangles: Related?

by Michael T. Battista, Douglas H. Clements, and Julie S. Meredith

Teacher: So, you knew you needed 90° turns to draw a square. You said that helped you know to use 90° when you drew this tilted rectangle. How come?

Cathy: Because a rectangle is just like a square but just longer, and all the sides are straight. Well, not straight, but not tilted like that (makes an acute angle with her hands). They're all like that (shows a right angle with her hands) and so are the squares.

Teacher: And that's 90 [showing hands put together at a 90° angle]?

Cathy: Yes. Really, a square *is* a rectangle.

Teacher: Does that make sense to you?

Cathy: It wouldn't to my [four-year-old] sister, but it sort of does to me.

Teacher: How would you explain it to her?

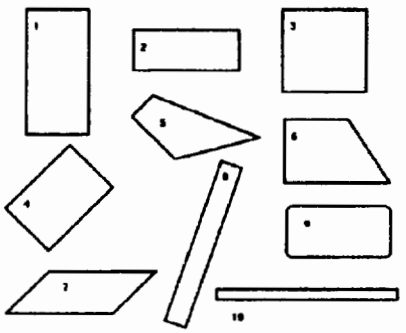
Cathy: We have these stretchy square bathroom things. And I'd tell her to stretch it out and it would be a rectangle.

To Cathy, a second grader, it "sort of made sense" that a square is a rectangle because a square could be stretched into a rectangle. She is using visual reasoning to relate one class of figures to another. Is this similar to the visual reasoning we saw kindergartners use in the previous column?

Actually, Cathy's reasoning may be more sophisticated than that. She had already demonstrated her knowledge that squares and rectangles are similar in having angles made by 90° turns. She may have understood intuitively that all rectangles could be generated from one another by certain "acceptable" transformations—ones that preserve 90° turns.

Let's look at several fifth graders, who are also struggling with similar questions. Jon was working on the square in the "Rectangle: What Can You Draw?" activity.

Directions: Which figures can you draw using a rectangle procedure with two inputs? You may turn before you draw a figure using RT or LT. Explain your reasoning (Battista & Clements, 1991).



Jon: This one [pointing to #3, the square] is not a rectangle. It's a square. It has equal sides.

Teacher: Can you do it with your rectangle procedure?

Jon: No, because the sides are equal. So that would be a "no."

Teacher: So, no matter what you tried, you couldn't make it with your rectangle procedure?

Jon: You couldn't, no, because the sides are equal.

Teacher: On your rectangle procedure, what does this first input stand for?

Jon: The 20? These sides.

Teacher: What does the 40 stand for?

Jon: Yeah, you *could* do it. If you put like 40, 40, 40, and 40 [again, motions].

Teacher: Ok, try it.

Jon: So that would be a square?

Teacher: Can you draw a square with your rectangle procedure?

Jon: You could draw it, but it wouldn't be a rectangle.

Even with prompting, Jon doesn't want to call the square a rectangle. In his view, you can draw a square with the rectangle procedure, but that doesn't make it a rectangle. Here is a dialogue with another fifth grader.

Teacher: Why do you think a square is not a rectangle, Jane?

Jane: Each side is equal to each other. But in a

rectangle there are two longer sides that equal each other and the other two sides equal each other but they're short.

We've all heard this before. Jane has simply described shapes she's used to calling rectangles. Because all of them have two long sides and two short sides, she includes this characteristic in her list of properties.

Teacher: How could you make a square with the RECT procedure?

Jane: You put in two equal numbers. And that's the distance [length] and the width. If they are the same amount, then it will come out to be a square.

Teacher: So it did come out to be a square? That is a square, you're telling me?

Jane: Yes, and a rectangle. But it's more a square, because we know it more as a square.

The second grader below tries to deal with the problem by inventing new language, much like one of the kindergartners whose work we discussed in the previous column.

Teacher: Is everything that RECT draws a rectangle?

Bob: That's [points to square on the screen] not a rectangle.

Teacher: How come?

Bob: Because the sides are the same size?

Teacher: So, this square [pointing to the square on the sheet] is not a rectangle?

Bob: I think it's a special kind of rectangle.

Teacher: So, is this [pointing to the square on the screen] a rectangle?

Bob: It's a special kind of rectangle.

Bob found a way to deal with the conflict of a square being drawn by a rectangle procedure. He invented a language that allowed him to avoid the uncomfortable statement "a square is a rectangle." Instead, he said that a square is a *special kind* of rectangle but not a rectangle.

Fifth graders tried to come to grips with the same question in a class discussion.

Lisa: I have a different question. Why can't we call squares equilateral rectangles?

Keith: A square classifies as a bunch of things. An equilateral rectangle doesn't classify as all the things that are square.

Teacher: Give me an example of a square that isn't an equilateral rectangle.

Keith: Well, like a diamond.

Teacher: [Draws one and has Keith clarify that he means a diamond with 90° turns. Keith still maintains that the drawing is not an equilateral rectangle.]

Lisa: All you have to do is turn it and it would be both a square and an equilateral rectangle in my definition.

Keith does not think a square and an equilateral rectangle are the same. He's on the road to thinking hierarchically, but has a way to go. Lisa, who does think with hierarchies, still uses visual thinking to support her argument. A bit later, the teacher throws out an additional challenge.

Teacher: Can our variable square procedure, SQUARE :X, be used to make a variable rectangle procedure?

Ken: No. There are two longer lines on a rectangle. They are longer than a square. All the lines are not equal in a rectangle; they are in a square. So if you think that, you can't draw a rectangle with a square procedure.

Peter: In the sense that the 10 or whatever you put down for the square represents all the sides, which wouldn't work because all the sides would be equal. So you'd have to make a new procedure for it.

Jacky: You have mentioned that opposite sides are parallel and equal. It's the same way with a square except that all sides are equal. So that the two sides that are parallel are still equal. So a square in the sense that you're saying is a still a rectangle, but a rectangle is not a square.

Teacher: Can we build any rectangle with the square procedure?

Jacky: Yes, you can.

Teacher: Can I build a rectangle with sides of 20 and 40?

Jacky: No, sorry. You can't build every single rectangle with the square procedure, but you can build one rectangle with the square procedure.

In pairs, students now move on to the "Rectangle: What Can You Draw?" activity. As they get to the square on the sheet, Jacky says, "It's a square." Peter illustrates his confusion over classification, saying, "A square can be a rectangle...wait. A rectangle can be a square but a square can't be a rectangle." Jacky starts to correct him:

"A square can be a rectangle." Peter interrupts, "Oh, yeah [laughing]."

All of these students see that the square procedure cannot be used to make rectangles. Jacky, however, is the only student who seems to understand the traditional mathematical perspective of classifying squares and rectangles. However, her comment "in the sense that you're saying" is suggestive. She has not yet accepted this organization as her *own*. The following episode further illustrates this point.

Teacher: If I typed in **rect 50 51**, what would it be [before hitting return]?

Peter: Probably about a square.

Jacky: A rectangle, but it wouldn't—

Peter: It would be a rectangle, but sorta like—

Jacky: It would be a rectangle, but it wouldn't be a perfect square. [They hit return.]

Jacky: You see, it's not a perfect square.

Peter: [Measures the top (longer) side with his fingers.] It's only one step off.

Even though Peter and Jacky say that the 50 51 rectangle is a rectangle and not a square, their language seems to indicate their belief in such a thing as an "imperfect square"—a figure that looks like a square but does not have all sides equal. They still put stock in their informal system, more so than the traditional logical classification system. They may adopt the traditional system only after many experiences of seeing not only its logic but its *benefits*. One advantage of Logo is that some of these benefits can be shown early on. For example, consider the simplicity of these two definitions:

```
to rectangle :length1 :length2
parallelogram :length1 :length2 90
end

to square :length
rectangle :length :length
end
```

Conclusions

All these vignettes share two features. First, activities are posed in ways that can promote geometric thinking. Second, the teacher is doing a *lot* of work, questioning, probing, clarifying, and challenging.

Given both of these features, this study shows that Logo can help students think of shapes in terms of their properties (Clements & Battista, 1992). Logo explorations of relationships between shapes such as squares and rectangles mean different things to different students. For some students, such as second-grader Cathy, such explorations extend their visual thinking. These students incorporate visual transformations that express their knowledge of the relationships. For several of the fifth graders, the explorations encouraged analysis and refinement of their definitions for shapes. The explorations helped other fifth graders understand the logical organization of properties into hierarchies, and—possibly—adopt it as their own.

References

Battista, M. T., & Clements, D. H. (1991). *Logo geometry*. Morristown, NJ: Silver Burdett & Ginn.

Clements, D. H., & Battista, M. T. (1992). *The development of a Logo-based elementary school geometry curriculum (Final Report: NSF Grant No.: MDR-8651668)*. Buffalo, NY/Kent, OH: State University of New York at Buffalo/Kent State University.

Preparation of this material was partially supported by the National Science Foundation under Grant No. MDR-8651668. Any opinions, findings, and conclusions or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the National Science Foundation.

Michael T. Battista is professor of mathematics education at Kent State University. He has conducted research and published in the areas of mathematics learning and computer applications in education.

Douglas H. Clements, associate professor at the State University of New York at Buffalo, has studied the use of Logo environments in developing children's creative, mathematics, metacognitive, problem-solving, and social abilities.

Julie S. Meredith is a mathematics education doctoral student at the State University of New York at Buffalo. She has taught secondary mathematics and computer science, gifted math at the middle school level, and mathematics methods courses. She is currently designing and programming Logo microworlds for the NSF-funded *Investigations* project.

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

Solving the Parallelogram of Forces Using *LogoWriter*

by Orlando Mihich

Extra for Experts

This article was written as a part of an Advanced Logo Seminar held in New York City in the spring of 1992. In the article, Orlando Mihich describes a set of Logo tools for solving vector addition problems, using both a mathematical technique involving sines and cosines and a Logo-oriented approach using multiple turtles. Mihich ends by suggesting how students can become "architect-programmers" and use these tools to explore the physics of buildings, bridges, and other structures."

I think it's important to note something about Mihich's work. His students are working in three different content areas simultaneously They're learning new techniques of Logo programming, they're learning about mathematics, and they're exploring architecture. Because Mihich's curriculum is integrated in this fashion, students are provided with a context for their learning, and this in turn gives them an understanding of why the assignments are what they are. As LX readers will note from previous articles in this column, I think this is a characteristic needed throughout our curriculum.—MAH

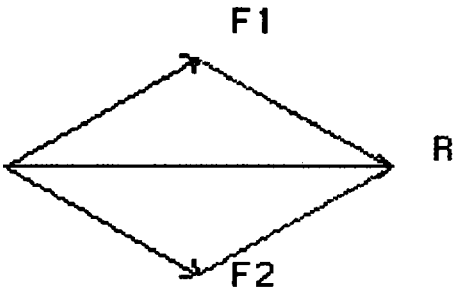
Introduction

Some physical quantities, such as length, volume, mass, and time, can be expressed in terms of magnitude alone, as single numbers. The height of a table is completely defined as 0.73 meters, and a carton of milk as 2 liters. These quantities, expressed completely by single numbers, are called scalars.

Other physical quantities, such as force, acceleration, and velocity, cannot be fully described in terms of magnitude alone. In addition to magnitude, these quantities have a specific direction. These quantities are called vectors. They are usually shown as arrows.

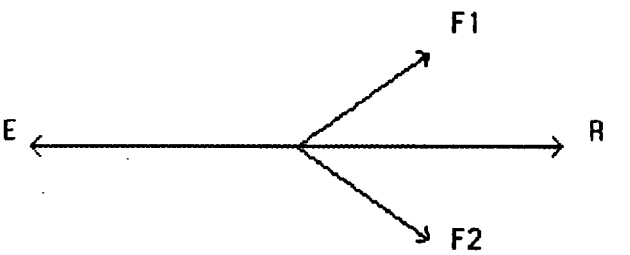
This vector could represent a wind blowing from 45 degrees west of south with a magnitude of 50 mph.

When two forces, F1 and F2, act on the same point at an angle other than 0 or 180 degrees, the resultant force, R, can be found by the parallelogram method.



An object under the effect of the two forces will move along the diagonal of the parallelogram of forces, that is, the resultant force R.

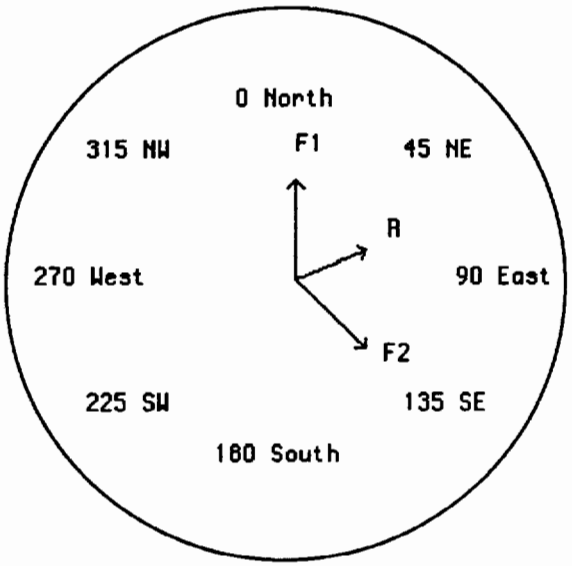
The object will remain in a state of equilibrium (i.e., unmoving) when an opposing force balances the effect of the resultant force R. The equilibrant force, E, must have the same magnitude as R but act in the opposite direction. For example, in the following figure the first number represents the magnitude, or the length of the line, of the vector; the second number represents the heading of the vector as indicated by the arrow.



F1	70	55
F2	70	125
R	115	90
E	115	270

In *LogoWriter*, 0 degrees is north (toward the top of the screen), 90 degrees is east (toward the right of the screen), 180 degrees is south (toward the bottom of the screen), and 270 degrees is west (toward the left of the screen).

F1	50	0
F2	70	135
R	38	68



There are three different approaches to solving these sorts of problems.

The Graphic Solution

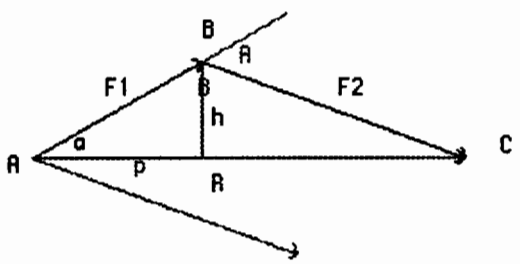
The resultant force can be found by drawing the parallelogram accurately to scale and measuring the magnitude of the diagonal with a ruler. At the junior high school level, students use graph paper, a ruler, and a protractor to find the magnitude of the resultant force. Results will vary slightly from student to student.

The Trigonometric Logo Solution

The resultant of two vectors acting at an acute or obtuse angle is usually found trigonometrically by the laws of sines and cosines. The equation for the resultant is:

$$R = \sqrt{F12 + F22 - 2F1F2 \cos 180 - A}$$

where A is the angle between the two forces F1 and F2.



Here,

$$F1^2 = h^2 + p^2$$

$$F2^2 = h^2 + (R - p)^2$$

$$F2^2 = F1^2 - p^2 + R^2 - 2Rp + p^2$$

$$F2^2 = F1^2 + R^2 - 2Rp$$

$$p = F1 \cos a$$

$$F2^2 = F1^2 + R^2 - 2\ R\ F1 \cos a$$

and similarly

$$R^2 = F1^2 + F2^2 - 2\ F1\ F2 \cos B$$

or

$$R^2 = F1^2 + F2^2 - 2\ F1\ F2 \cos 180 - A$$

The Logo resultant for the two vectors, F1 magnitude1 direction1, and F2 magnitude2 direction2, is:

```
to resultant :F1 :F2
output sqrt ((sq first :F1) + (sq
   first :F2))- (2 * (first :F1) *
   (first :F2)) * cos 180 - ((last
   :F1) - (last :F2))
end
```

The *LogoWriter* Solution.

The junior high school student can also write a program to draw the two forces and the resultant and find the values for magnitude and heading without knowing the laws of sines and cosines. The following program will draw the two forces, the resultant, and the equilibrant force:

```
to startup
rg
ht
ct
cc
tell [0 1 2]
pu
setpos [0 0]
make "start [0 0]
type [Introduce magnitude and direc-
   tion for]
type char 13
type [Vector1]
type char 32
end
```

```
to arrow
right 45
back 5
forward 5
left 90
back 5
forward 5
right 45
end

to vector1 :magnitude1 :direction1
make "m1 :magnitude1
make "d1 :direction1
tell 0
ht
setc 2
seth :direction1
pd
forward :magnitude1
arrow
pu
forward 12
pd
label [F1]
pu
back 12
pd
pr (se [F1] :magnitude1 :direction1)
type [Introduce magnitude and direc-
   tion for]
type char 13
type [Vector2]
type char 32
end

to vector2 :magnitude2 :direction2
make "m2 :magnitude2
make "d2 :direction2
tell 1
ht
setc 2
seth :direction2
pd
forward :magnitude2
arrow
pu
forward 12
pd
label [F2]
pu
back 12
back :magnitude2
pd
print (sentence [F2] :magnitude2
   :direction2)
cc
type [Press the return key.]
type char 13
type [Resultant]
end

to resultant
tell 0
seth :d2
pu
forward :m2
make "p pos
tell 2
setc 5
seth towards ask 0 [pos]
make "h heading
pd
setpos :p
arrow
pu
forward 12
pd
label [R]
pu
back 12
pu
setpos :start
make "d distance ask 0 [pos]
print (sentence [R] round :d round
   :h)
cc
type [Press the return key.]
type char 13
type [Equilibrant]
end

to equilibrant
tell 2
seth (:h + 180)
pd
forward :d
arrow
pu
forward 12
pd
label [E]
pu
back 12
print (sentence [E] round distance
   ask 1 [pos] round heading)
cc
type [Type a key to restart.]
type char 13
make "key readchar
startup
end
```

The program requests the magnitude and direction of vector1 and then vector2. Typing the two numbers for Vector1, for example, 50 for magnitude and 45 for direction, will draw a line of 50 turtle steps (i.e., 50 newtons, a unit of force), and set the turtle's heading at 45 degrees (NE). After the second vector is introduced, the resultant, R, is the diagonal vector connecting the point of origin of the two vectors and the opposite corner of the parallelogram. Since opposite sides of the parallelogram are equal, the turtle in vector1 draws vector1 and then, in resultant, follows the instructions for magnitude and direction of vector2 and "makes a position" for drawing the resultant. The resultant's magnitude and heading are obtained by asking the turtle in the opposite corner of the parallelogram for the respective values. The equilibrant force is identical in magnitude to the resultant force only heading in the opposite direction (heading + 180).

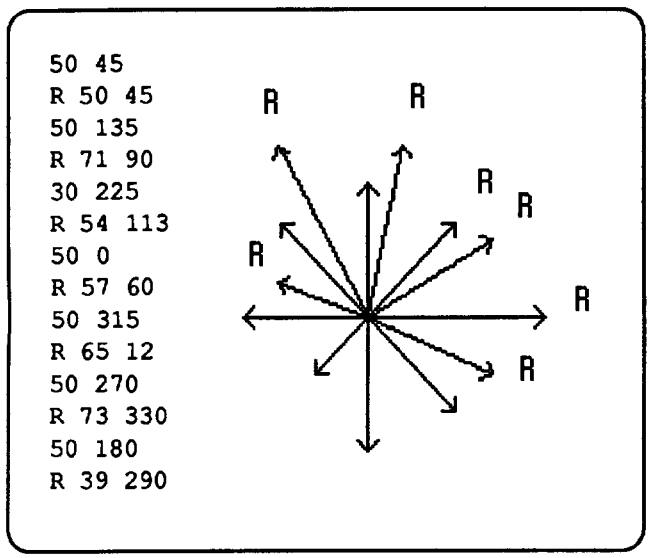
Handling Multiple Vectors

There are two ways of finding the resultant of two or more vectors acting at the same point.

Technique 1. Using the parallelogram method, the resultant of the first two vectors is used with a third vector to find the second resultant, the second resultant is used with a fourth vector to find the third resultant, and so on, until the last vector is introduced and the final resultant is obtained. The following program starts with a vector of 0 magnitude and 0 direction (make "r [0 0]); therefore, the first typed vector is also the first resultant.

```
to startup
rg
ht
ct
cc
make "start [0 0]
make "r [0 0]
v
end

to arrow
right 45
back 5
forward 5
left 90
back 5
forward 5
right 45
end

to v
cc
type [Type magnitude and direction
   for vector.]
type char 13
make "v readlistcc
setc 2
seth (last :v)
pd
forward (first :v)
arrow
back (first :v)
print (sentence (first :v) (last
   :v))
r
end

to r
seth (last :r)
pu
forward (first :r)
seth (last :v)
forward (first :v)
make "p pos
pu
setpos :start
seth towards :p
make "h heading
setc 5
pd
setpos :p
arrow
pu
forward 12
pd
label [R]
pu
setpos :start
pd
print (sentence [R] round resultant
   :r :v round :h)
make "r list round resultant :r :v
   round :h
v
end

to resultant :r :v
op sqrt ((sq first :r) + (sq first
   :v))(2 * (first :r) * (first :v))
   * cos 180 - ((last :r) - (last
   :v))
end

to sq :n
op :n * :n
end
```

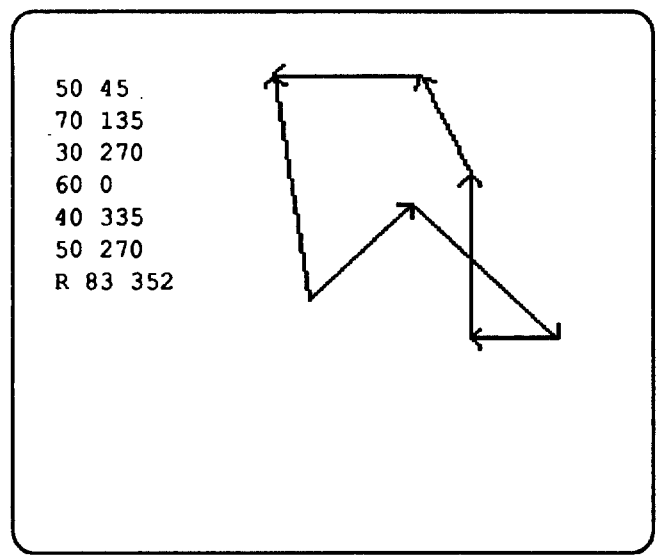
This produces a multiple-vectors screen:



Technique 2. A shortcut for the parallelogram of forces is to place vectors head to tail. The resultant force, R, is a straight line connecting the tail of the first vector with the head of the second vector. The orientation of the resultant is toward the head of the second vector.

```
to startup
rg
ht
ct
cc
setup
print [To add a vector type V, and
   introduce magnitude]
print [and direction, then type R.]
end

to setup
tell [0 1]
ht
pu
setpos [0 0]
end

to arrow
right 45
back 5
forward 5
left 90
back 5
forward 5
right 45
end

to v :m :d
tell 0 setc 2
seth :d
pd
forward :m
arrow
print (sentence :m :d)
end

to r
tell 0
make "p pos
tell 1
make "o pos
seth towards ask 0 [pos]
setc 5
pd
setpos :p
arrow
pu
setpos :o
print (sentence [R] round magnitude
   round direction)
end

to magnitude
tell 1
output distance ask 0 [pos]
end

to direction
tell 1
op heading
end
```

This produces the following screen:



Vectors in the Classroom

In the classroom, students explore forces by engaging in various activities, for example, pulling a heavy object like the teacher's desk. The desk moves following the resultant force, and eventually students discover that the closer they stand together, the easier it is to pull the desk. In another activity, two students pull a rope at the ends while a third student in the middle resists the pull. This student discovers that as the angle between the two students pulling the rope increases, the easier it gets to resist the pull and maintain equilibrium.

In the class "Why Buildings Stand Up," students engage in many similar activities. This class deals with the real environment; students use their intellectual and manual skills to learn about and create an environment with which they are familiar: the buildings and bridges surrounding them in everyday life. Students also love to work on computers, and by using *LogoWriter* they become the "architect-programmer." They teach the computer to show how forces act on structures, they write programs showing forces in equilibrium and the effect of wind forces on buildings, they generate screens to show tension and compression forces in a truss bridge, and so on. The following is one example of such activities.

The Truss Bridge. By holding hands firmly and pulling, students experience tension "first hand." When they push at each other without bending their arms, they "feel" compression. Tension is also demonstrated by stretching a rubber band. The increased length shows that tension lengthens. Squeezing a sponge shows that compression shortens.

Trusses are structures made out of bars that work either in tension or compression without bending. A truss bridge is made of triangular trusses. Students build model bridges by using tongue depressors; then they determine which bar is in tension or compression by substituting a string for the depressor. The string in tension becomes a straight line. With these experiences and the Logo vector tools described previously, students can analyze bridges, such as the one shown below, and then move to building working models.

References

Apsen, Boris. (1978). *Repetitorij Vise Matematike.* Tehnicka Knjiga Zagreb.

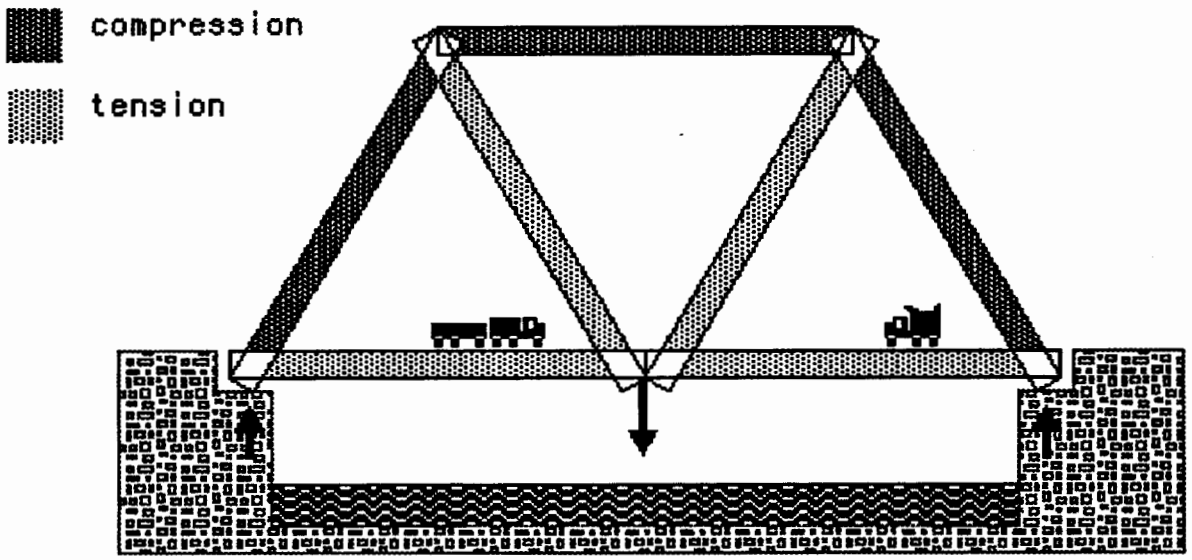
Lewis, Philip G. (1990). *Approaching Precalculus Mathematics Discretely.* MIT Press.

Salvadori, Mario. (1979). *Building—From Caves to Skyscrapers.* The Salvadori Educational Center on the Built Environment.

Tempel, Michael. (1992). Advanced Logo Seminar Handouts.

Trinklein, Frederick. (1990). *Modern Physics.* Holt, Rinehart and Winston, Inc.

Orlando Mihich
339 Pacific Avenue
Jersey City, NJ 07304



Global Logo Comments

by Dennis Harper

Logo Exchange Continental Editors

Africa	Asia	Australia	Europe	Latin America
Fatimata Seye Sylla	Marie Tada	Anne McDougall	Harry Pinxteren	José Valente
UNESCO/BREDA	St. Mary's Int. Sch.	Monash Univ.	Logo Centrum Nederland	NIED
BP 3311 Dakar	6-19 Seta 1-Chome	6 Riverside Dr.	P.O. Box 1408	UNICAMP
Senegal, West Africa	Setagaya-Ku	East Kew 3120	BK Nijmegen 6501	13082 Campinas
	Tokyo 158, Japan	Victoria, Australia	Netherlands	São Paulo, Brazil

In this column, we'll take a look at the state of Logo in Japan. Marie Tada interviews her *LX* Asian correspondence predecessor Hillel Weintraub, who is now back teaching in Japan after a couple years at MIT and Harvard. This interview provides some thoughts on education—and Logo—in Japan.

Marie: Hillel, you've been one of the people in Japan who was an early proponent of Logo, especially through the group that you founded in 1982, S.M.I.L.E., the Society for Microcomputing in Life and Education.

Hillel: Yes, I looked upon Logo as one possible way to affect the future directions of Japanese education, and I worked together with a number of other people who had similar hopes, specifically Hiroyoshi Goto (then of Uny Bynas and now of Fukutake Shoten's New Media Group), Teru Miyama (of Data Pop, a software house) and Masahiro Nakauye (then working in an education office and now a professor at a women's college in the Kobe area.) S.M.I.L.E. came into contact with, learned from, and hopefully gave some ideas to many of the men and women who are leaders in the education and technology field in Japan today.

Marie: What exactly was S.M.I.L.E.?

Hillel: Well, it wasn't "exactly" anything! In fact in was quite an in-exact organization. We were always rediscovering what we were about and therefore it was very hard to define us. That was our strength and of course, at the same time, our weakness. The fact that we didn't spend a lot of energy defining ourselves—something that is necessary to be successful sellers of one's self or group—meant that we never grew very much. Our largest meeting was 150 people, I think, but we published some interesting collections of ideas and our meetings were quite stimulating. Of course there were some central ideas that we believed in and acted upon in a variety of ways: (1) that education needed to be empowering for all children, (2) that science and technology were too important to leave in the hands of the scientists and technocrats and that we had to find ways to bring those who felt alienated from school math and science into the picture, and (3) that some of the boundaries, such as the separation between the arts and sciences or, more generally, between living and learning, or between life outside school and life within school—had to broken down. Our members were all influenced by Papert's *Mindstorms*, partially because S.M.I.L.E. bought a hundred or so copies of the book in Japanese and English and gave it to all new members to read! I think what stimulated us the most was the realization that we were on a threshold of sorts and really could influence the direction that education might take.

Marie: And did this turn out to be true?

Hillel: Well, "yes" and "no." I mean, we're always on a threshold, aren't we? In *Mindstorms* , Papert used the example of how, if careful thinking had been done in the early years after cars were invented, our roads and cities and pollution problems would be quite different now. This was quite impressive because we did see ourselves at a kind of crossroads. Am I mixing metaphors: "crossroads" and "thresholds"?

Marie: They're close enough, aren't they?

Hillel: I think so. Anyway, I think a number of us early Logo-ites and S.M.I.L.E.-ers see that while technical things are changing much faster these days, real changes still take time.

Yet I think we are still a positive group and believe that we have had and will continue to have an effect on Japanese education.

Marie: And the "no" part?

Hillel: Well, "no" in that all of this is very subtle. You know, it's really difficult to see anything very dramatic, but there are these little happenings all around Japan, like teachers here and there doing some innovative exploratory learning projects with their students, or an influential professor here and another there, saying something that maybe we were talking about 10 years ago at our meetings. But now the people nodding in agreement have more power to effect changes than we did.

Marie: Does S.M.I.L.E. still exist?

Hillel: Oh, yes, we're still an international member of ISTE, though I can never get them to refer to us as S.M.I.L.E. rather than the Society of Microcomputing in Life and Education! And the group in Tokyo has kind of died and all the members (though they will never stop being S.M.I.L.E.-ers and we still get together occasionally) have really gone on to other things. The group in Kyoto-Osaka-Kobe is still active, though many of us are giving a lot of energy to a new national organization called A-GENE (Association of Global Electronic Networking Educators), whose central idea is that telecommunications can help us effect important educational changes.

Marie: So you feel some satisfaction about what you've done and where things are heading?

Hillel: Well, sometimes I do, but when I hear you say it or when I become aware of feeling satisfaction, I think I'd better wake up. I mean, we really can't be smug about anything, can we? Because we really know so little about how to design decent human societies. I'm hoping to enter into a project working with 35 seventh graders in which we'll all have our own notebook computers and explore for two to three years how this affects the way we relate to each other and to how and what we notice, note, and know. It would be great to have the opportunity to try this out, but sometimes I wonder about the possibility that experiments like this could lead to a further gap between those who are materially well off in our world and those who aren't. I'm also worried that because the computer is a powerful material object that it will further entrench us in the already strong belief that we need material things to be happy, to be educated, and so on. There is a growing number of people, and I try to be one of them as much as I can, who think we need to spend a little more energy in getting the "haves" to examine their society's predominant values: that life may be more than having a pair of Nikes and a neat stereo plus a television so we can learn that our new Nikes and stereo (not to mention the television itself!) aren't the latest thing any more.

Marie: So one of the purposes of your project is to undercut the importance of technology, while using technology to a greater extent than ever before?

Hillel: Yes, that's a good way of putting it. We can't put technology in its proper perspective by hiding from it. I want to use it to help us learn to have confidence in our own voices so that we stay in control, rather than being controlled—that is certainly a central concern for me.

Marie: You were in Japan at the beginning of this movement for a new education through technology. Then in 1988 you went back to the U.S. to attend Harvard's Graduate School of Education and study and work with Papert's group at the Media Lab. How did those three years affect you?

Hillel: Wow! In 20 words or less, right?

Marie: Well, you know, being with Papert's Logo group at MIT is a kind of dream that many people in the Logo world have had at one time or another.

Hillel: One thing those three years made me realize was how rare "real" education happens in school contexts. By "real," I mean learners communicating with each other and creating new meaning together. Being at Harvard's Ed School was a great experience because of the interaction I had with other students, but most of the professors that I came into contact with there were so involved in their own ideas that they didn't have time to remember what the purpose of teaching was—not to further their own personal theories but to help their students develop their own ideas. Actually, Joe Maxwell, an ethnographer, was one person I found consistently interested in hearing the ideas of others and of valuing their experiences. After all the average age of grad stu-

dents in the Ed School was almost 40, and they were coming from all kinds of places and experiences! Most of them had more understanding and experience of educational situations than did the professors at Harvard, but few professors bothered to listen to them. I guess I can express the problem in a humorous paradoxical way: many of the professors there were so busy explaining their theories about why teachers needed to empower (that was a popular word!) their students and give credence to their multicultural voices (that was another popular expression) that they never had time to do this in the reality of their own classes or teaching.

Marie: What about the Media Lab?

Hillel: My experience there was only with the Epistemology and Learning Group (E&L), the part of the Media Lab led by Papert and Edith Ackermann. It was much more poignant than what I experienced within the walls of the Ed School. It was totally imperfect and marvelous. A group of humans struggling to communicate in a world so complex as to constantly upset communication. In a small way, for each of the 15 or so members of the E&L group, I was able to feel their humanity and struggle to make sense out of some aspect of learning and technology. At Harvard I had the idea that there were all these little communities that I could join if I wanted to be some professor's groupie. But at the Media Lab, that notion quickly became part of the dialog of trying to create a community of learners.

Marie: So was there and/or is there a community of learners there?

Hillel: Some people (including ones much closer to the situation than I ever was) would laugh at the notion, but I would say yes (and no, of course), because you know the notion of "community of learners" is just a bunch of words—I mean, it doesn't exist except in someone's article or thesis or mind. But in reality it's a fuzzy, imperfect notion. But at least in the E&L group, I think this struggle was recognized as such; it was a part of the conversation. At Harvard, I couldn't imagine students (for example) in a Theories of Intelligence course challenging the professor's dictatorial approach to knowing. But for me, that's the most important thing to be talking about, not x or y theory of intelligence.

But listen, I have to say honestly that I wasn't totally involved at either place, and I didn't feel that my life depended upon my finishing my doctoral work. I was just a stranger passing through. That's part of the weakness of my own character and defined the experience I had. Basically, I see myself as a teacher, and I saw my graduate work as a means of making myself more effective in another environment. I know that many people have had different experiences that are, of course, just as valid as mine. I do want to say that at MIT and the Media Lab, I felt listened to and valued much more than I did at Harvard, which was strange because I'm such an atypical person to even be walking on the MIT campus.

Marie: Can you say a bit more about the effect that experience had on you?

Hillel: I think most of the effects are things that I couldn't enumerate, but contact with a few people like Carol Sperry, who ran Project Mindstorm in California, and some of my fellow grad students at the Ed School and in the E&L group have left various marks on my consciousness. Edith Ackermann was a rare professor who listened to what I had to offer and helped me build on that. The best course during my three years was a Design Seminar I took at MIT with her and Don Schon (Dean of Urban Studies Program) and Bill Porter (Dean of the School of Architecture), all of whom listened to each other as well as to the 12 graduate students or visiting professors who joined that seminar. It's truly sad how rare that experience was, when it should be what happens in all classes—the voice of every learner was equally respected. I did have three classes at the Ed School where this was the case—with Joe Maxwell's ethnographic research classes, with Catherine Krupnick in a class on the use of video in working with teachers, and with Eleanor Duckworth in her course about teaching and learning. The problems for me came as it got closer to doing so-called official doctoral work. Then more pressure was put on me to think in particular ways, rather than for anyone to listen to the ways that I thought or to help me build on those thoughts.

Marie: But now that you're back in Japan, I wonder what you're thinking. Can you notice any changes since you've been away?

Hillel: I think that the Japanese as a whole are talking more about internationalization and helping people develop their creativity and independent thinking. This is positive, though of course it really will take years to begin to affect what happens in the schools in any noticeable way. However, I must say that my school administrators are more supportive of my ideas now than they were before I left.

Marie: To what do you attribute that?

Hillel: Time, and maybe the power of the names of Harvard and MIT.

Marie: And what is it that you're telling them or doing?

Hillel: Well, I've been talking about the importance of "meaning making" over "meaning taking" and we've actually created a room we are calling our Cooperative Design Center. It's a room that has been designed in a way to encourage learners to make meaning together. So I've been talking a lot about the responsibility of teachers to design particular kinds of environments. A few people are listening. Not many yet. My school, Doshisha International Junior and Senior High School, located in Kyoto, is a school run by Japanese and designed to educate Japanese students who have lived abroad. We are not a typical foreign-run International School for non-Japanese living in Japan. Therefore, the steps that are being taken by our administration are a big, positive step forward in introducing these kinds of ideas within the Japanese school structure.

Marie: So what's your final word for us educa tors?

Hillel: "Final Word!" Oh, it sounds so, so ... final. It reminds me of when people would ask me, "Are you going to stay in Japan forever?" or "Will you always do or be xxxxx?" Forever? Always? Final? I believe that the purpose of having an idea is just to help us have the next one. But of course thinking this won't stop me from saying something "very important": we have to keep reminding each other that the answer to improving education—and, thus, life for human beings—doesn't lie with the number or kind of computers (one per class or one per child, Apple or IBM or NEC) or in the software used (Logo, *HyperCard*, multimedia, word processing) but in something called *relationship* and how people, specifically teachers in traditional positions of power in schools, relate to children and help them relate to each other and to knowledge. The question that I want to explore over the next few years with my co-teachers and learners is how can we design learning environments in which technology will help us empower each other; how can we give a new definition to learning as meaning-making in a community; how can we begin to recognize each other's voice as an important part of this community meaning-making?

Marie: These are part of the Logo philosophy, aren't they?

Hillel: Yes, I believe that we're always building on fundamental, powerful educational ideas as we go spiraling off into the future!

Greg Harris and Bruce Yang, third graders, became interested in gravity. Their hypermedia presentation included animation.

Logo Exchange

ISTE • 1787 Agate Street • Eugene, OR 97403-1923 • InterNet: ISTE @ Oregon.uoregon.edu
503/346-4414 • ISTE Order Desk : 800/336-5191

SIGLogo
ISTE
1787 Agate Street
Eugene, OR 97403-1923