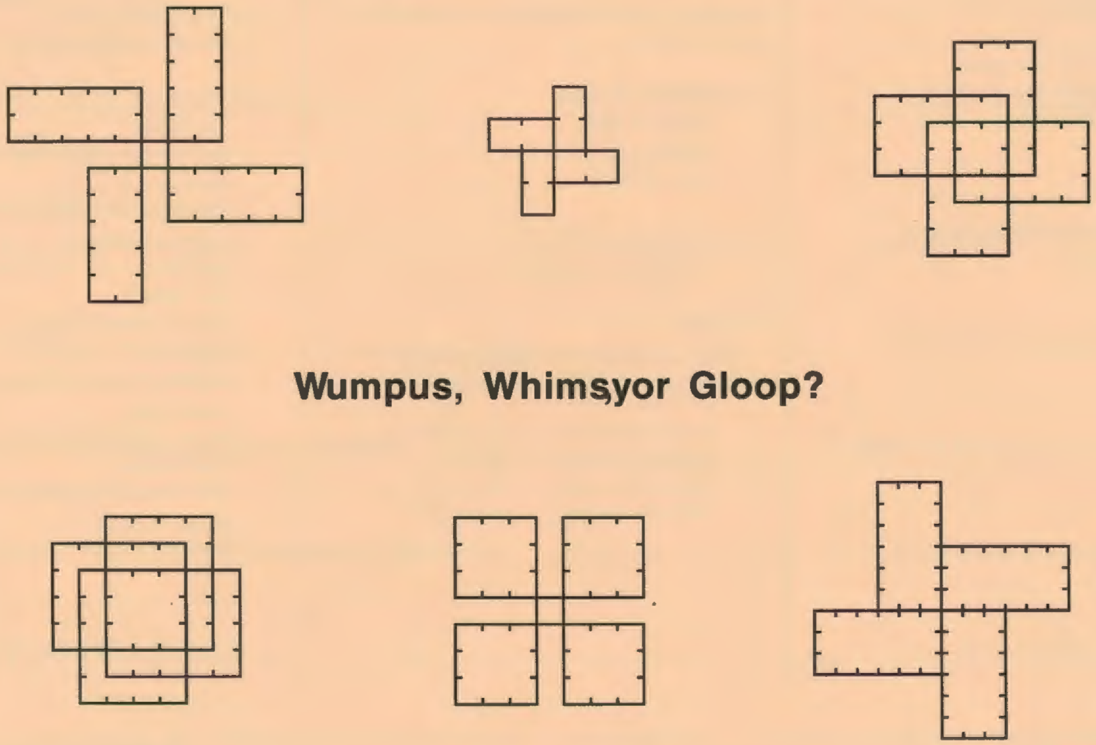
LOGO EXCHANGE

Volume 15 Number 1 Fall 1996

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



Editorial Publisher
International Society for Technology in Education

Editor-in-Chief
Dorothy M. Fitch

Copy Editor
Jennifer Roland

Founding Editor
Tom Lough

International Editor
Dennis Harper

SIGLogo Officers
Chuck Friesen, President
Steve Sesko, Vice-President
Hope Chafiian, Secretary/Treasurer
Dorothy Fitch, Editor

Production
Kerry Lutz

Director of Advertising Services
David Shuey

Submission of Manuscripts

Logo Exchange is published quarterly by the International Society for Technology in Education Special Interest Group for Logo-Using Educators. *Logo Exchange* solicits articles on all aspects of Logo use in education.

Manuscripts should be sent by surface mail on a 3.5" disk (where possible). Preferred format is *Microsoft Word* for the Macintosh. ASCII files in either Macintosh or DOS format are also welcome. Submissions may be made by electronic mail as well. Where possible, graphics should also be submitted electronically. Please include electronic copy, either on disk (preferred) or by electronic mail, with any paper submissions. Paper submissions may be submitted for review if electronic copies are supplied upon acceptance.

Send surface mail to:
Dorothy M. Fitch
3 Derby Road
Derry, NH 03038

Send electronic mail to:
71760.366@compuserve.com

Deadlines

To be considered for publication, manuscripts must be received by the dates indicated below.

Volume 15, Number 3	Oct. 1, 1996
Volume 15, Number 4	Jan. 1, 1997
Volume 16, Number 1	Mar. 1, 1997
Volume 16, Number 2	June 1, 1997

Logo Exchange is published quarterly by the International Society for Technology in Education (ISTE), 1787 Agate Street, Eugene, OR 97403-1923, USA; 800/336-5191. This publication was produced using *Aldus PageMaker®*.

ISTE Members may join SIG/Logo for $24.00. Dues include a subscription to *Logo Exchange*. Non ISTE Member subscription rate is $34.00. Add $10 for mailing outside the USA. Send membership dues to ISTE. Add $4.00 for processing if payment does not accompany your dues. VISA, Mastercard, and Discover accepted.

Advertising space in *Logo Exchange* is limited. Please contact ISTE's director of advertising services for space availability and details.

Logo Exchange solicits articles on all topics of interest to Logo-using educators. Submission guidelines can be obtained by contacting the editor. 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 *Logo Exchange*, ISTE, 480 Charnelton St., Eugene, OR 97401-2626 USA. Periodicals postage paid at Eugene, OR. USPS# 660-130. ISTE is a nonprofit organization with its main offices housed at the University of Oregon. ISSN# 0888-6970

Logo Exchange

Volume 15 Number 1 Journal of the ISTE Special Interest Group for Logo-Using Educators Fall 1996

Contents

From the Editor

by Dorothy M. Fitch

Welcome back!

Welcome to another year of *Logo Exchange*. This issue marks the beginning of the 15th year of this journal. I hope that you will continue to find ideas that interest you and projects to try with your students in each issue.

Patterns and motion

In this fall's issue, you can explore patterns and motion in many different ways.

Ihor Charischak challenges us to find patterns in the designs made by a simple spirolateral program. What can you and your students discover about wumpuses, whimsies, and gloops?

Nancy Flynn invites us to explore Newton's first law of motion and sine waves as we study physics using the turtle.

This leads quite naturally into ***Robert Macdonald***'s mathematical exploration of the patterns created when a billiard ball bounces around a pool table.

Jim Muller, who finds ways to use Logo to explore just about anything, gets beginners moving around the Logo screen; then we all can travel about in a geography game.

Tom Lough, who was choosen to carry the Olympic flame in Connecticut, shares this exciting event with us and makes us feel as though we were right there with him.

Don Ryoti shows how he introduces recursion to his math education students. Because many Logo users find recursion a difficult concept, it seems appropriate to present many different approaches to help us understand it.

Doug Clements and ***Julie Sarama*** show us a plan for implementing change.

Now it's your turn!

I invite and encourage you to submit your Logo projects, ideas, and teaching strategies for publication in *LX*. What are you doing with your students that excites them (and you)? What do you do that works? If you don't feel particularly talented in communicating your ideas and projects to others, I can help you. The main thing is to get started!

I hear you saying, "But I don't have the time!" Perhaps that is true. But do you have a student or a group of students who does? Would they like to share what they do with Logo in your classroom? They could describe how they came up with the idea for their project, what obstacles they encountered, how they solved them, and what they learned and discovered along the way.

They could write about a favorite Logo lesson, a robotics project, a simulation that they developed, an activity that helped them understand Logo a bit better, or how they are communicating with students in another part of the world about Logo. Be sure they include any graphics they want to share!

So, students and teachers, let me know what you are doing, how you are using Logo, and what excites you about Logo. Please share your ideas with the rest of our readers!

Here's how you get started!

Send me your article (or just a small bit of it if you want to make sure you are on the right track) by e-mail, mail, or fax (see addresses and numbers at the end of this column).

Please send your article in ASCII text format or in a Microsoft Word file for the Mac (v. 4.0) or Windows (v. 6.0). You can also send a PageMaker file for either Mac or PC.

If you have any questions, feel free to call me at 603/425-2010. I look forward to hearing from you and your students this year.

Happy Logo adventures! ▲

Dorothy M. Fitch
Editor, *LX*
3 Derby Road
Derry, NH 03038
Telephone: 603/425-2010
Fax: 603/425-6487
E-mail: 71760.366@compuserve.com

Quarterly Quantum
Carry :torch Pass :flame

by Tom Lough

Carrying the Olympic flame is a moving event, in every sense of the word! On June 16th (Father's Day, no less!) I ran about one-half mile through Hartford, Connecticut, as part of the 1996 Olympic Torch Relay, holding aloft a beautifully designed and engineered torch with a flame that had originated on Mount Olympus. What a thrill!

Later, someone asked me what part of the relay made the greatest impression on me. My immediate reaction was, "Running with the lighted torch!" After all, this is the activity that took up most of the all-too-short period of time.

But, upon later reflection, I realized that there was another part of the relay that affected me much more profoundly. I suppose it did not come readily to mind because it happened so quickly. But in that brief moment, I remember experiencing a powerful emotion.

The event I am referring to is the passing of the flame. You see, the torch relay is not about carrying a torch; it is about moving the flame.

The mission of the relay was to deliver the Olympic flame to the opening ceremonies in Atlanta on July 19 in a manner that allowed as many people as possible to see it. And few came closer to the flame than those who carried the torches. Let me tell you about my encounter.

Runners for each segment of the relay were carried on ahead of the flame by a special van and deposited at designated transfer points. When I stepped out of the van with my unlighted torch, the crowd on the street began to stir. They knew that their vigil was nearly over. I could feel the excitement start to build!

After a couple of minutes, I heard the sound of sirens and applause from afar. Then the caravan of flashing lights came into view. There! In the middle of the hubbub I saw the white-uniformed figure of the runner, Ann Smith of Johnstown, Pennsylvania, holding the flaming torch aloft!

At about that same instant, a technician on a motorcycle roared up and turned on the gas supply for my torch. I could hear the energetic hissing as the gas escaped the burner in search of a flame.

In a few moments, Ann reached my position and paused a couple of meters away. Slowly and dramatically, she raised her flaming firebrand up into the air and then extended it forward. In response, I raised my own torch and moved it toward hers.

Whoooooosh!

The flame leaped across the gap between us, completing the arch we formed and engulfing the top of my torch. What a moment! It was simply beautiful!

The run was a blur, even though I tried to go slowly and show the flame to everyone along the way. All too soon, I saw the next runner, Bob Mayo of Granby, Connecticut, up ahead, waiting with his unlit torch.

As the flame leaped eagerly from my torch to his, I could feel a surge of the Olympic spirit passing through me to him as well. It was these moments of passage that made a deep impression on me.

I later realized that I had experienced a similar feeling several times before: when a Logo student I was helping suddenly smiled and surged on ahead, and when a Logo teacher I was assisting began to get excited about what this computer language could empower her students to do.

All of us are carrying the torch for Logo as we go about our day-to-day teaching. But it is also important to realize that we must pass the flame on as well. The embodied spirit of Logo—the excitement and thrill of this special type of learning—must be an integral part of our teaching activity.

Even though we each pass the Logo flame to others in different ways, let's become more aware of these special moments. It is in these moments that lives are changed in ways we cannot imagine.

And isn't that what teaching is all about? ▲

FD 100!

Tom Lough
Founding Editor
PO Box 394
Simsbury, CT 06070
E-mail: 70020.223@compuserve.com

Experiment: Online code!

This year, most of the programs printed in *LX* will also be available electronically. This should save you lots of time (no more entering long programs) and help prevent bugs (the kind that creeps in through typing errors).

The programs will be exactly as listed in *LX*. However, if any readers would like to convert the programs to other versions of Logo and send them back to me electronciially, I can then make them available to others on request.

In this issue, the following Logo programs are available electronically:

- **SPIROS**
  from "Wumpus, Whimsy, or Gloop?"
- **POOLBALL**
  from "The Mystery of the Moving Billiard Ball"
- **PHYSICS**
  from "Logo in Motion"
- **TEXAS**
  from "Exploring Turtle Geography"

To receive one or more of these listings electronically, send a request to me at 71760.366@compuserve.com with the name of the program(s) you would like to receive (use the bold name listed above). Also indicate which version of Logo you are using. If that specific version is available, you will receive it; otherwise you will receive the version listed in *LX*.

I hope you will find this new service useful. As always, I welcome your comments and suggestions.

Dorothy Fitch, *LX* editor
E-mail: 71760.366@compuserve.com

Logosium '96

reported by Theodore Norton

Logosium lives! On June 14, 1996, Marian Rosen and Michael Tempel opened the third annual Logosium, sponsored by the ISTE SIGLogo and The Logo Foundation, in an atmosphere of cordiality created by members and friends of the St. Paul Logo Project. Logosium was hosted by the St. Anthony Park Elementary School in St. Paul, Minnesota, an inviting site that is well endowed with computer resources.

The 85 participants were drawn from the local area, from the rest of the country, and from around the world (Mexico, Canada, Brazil, Australia, Saudi Arabia, and Southeast Asia). There were four sessions with five to six topics each, including open labs. The topics, together with their student and adult facilitators, included:

- *We're Right, They're Wrong* with Gary Stager
- *Logo-Based Multimedia Projects* with Sharnee Chait and Dorothy Fitch
- *Logo on the World Wide Web* with Michael Tempel
- *Expert Mathematician—Power Math Logo* with Jim Baker and Dale Hulme
- *Getting Emotional With Lego & Logo* with Keith Braafladt and Natalie Rusk
- *Game Design & Construction—Teachers* with Hope Chafiian and Michael Tempel
- *Game Design & Construction—Kids* with Jake Bruer and Michael Petersen
- *Visual Modeling With Logo* with Darrell Mohrhauser
- *Maya Quest: Logo With CDs & Videodiscs* with Kathy Ames and Greg Anderson
- *Logo & ESL* with Charles Lee
- *Logo & Robotics* with Dorothy Fitch and Paul Krocheski
- *Logo Supports Formal Geometry and...* with Kassidy Newscom, Barbara Sylvester, Breana Sylvester, and Danielle Wells
- *...Not So Formal Multicultural Geometry* with Nancy Flynn
- *MicroWorlds Projects About Inventions* with Eleanore Bednarsh and Joan Nelson
- *The Nitty Gritty of Importing Graphics* with Ron Beck
- *The Programmable Brick* with Michael Tempel and David Tempel
- *Logo in an Elementary School* with Charlotte Coan and Judy Steiner
- *Logo Animation Techniques* with Bill Spezeski
- *Logo and Laptops—When is Enough?* with Diane Jackson, Paul Krocheski, and Marian Rosen
- *Myst & MicroWorlds* with Keith Braafladt and Natalie Rusk

As this list suggests, a wide range of interests, from telecommunications to "living" computerized environments, was represented, as were different Logo-based systems: MicroWorlds was put through its parallel processing paces, while Terrapin and PC Logo-driven robots whizzed and whirled across laboratory floors. Children made exciting and articulate presentations, as their mentors pondered Logo's future.

A high point of the day was the keynote address delivered by Geraldine Kozberg, founder of the St. Paul Logo Project, on "Whatever Happened to the Revolution?" Ms. Kozberg recalled the heyday of "classical" Logo in 1984–1986: Brian Harvey's posing of her title question in 1986 and her reply, "We are a culture in transition, not a revolution." Ten years later, according to Ms. Kozberg, the nonrevolutionary character of the Logo movement has been confirmed. However, the problems of the present are posed not by technology but by the denial of social justice. Once again we must build the social and behavioral prerequisites for effective learning for all children, even as its context is lost to technocentric commerce. Accordingly, the emphasis of the next 15 years must be placed on equity and the dynamics of learning environments.

Next year, Logosium is again planned to take place in conjunction with NECC, this time in Seattle, Washington. Watch for information about Logosium '97 in future issues of *LX*. ▲

The Beginner's Corner: Exploring the Logo Screen

by Jim Muller

As we begin another year, it might be useful to review some of the ways we can help Logo novices get excited about Logo. Practice in the basic commands is essential. But how can you keep this exciting? Hopefully you will find a new idea or two in this article on "getting started."

Getting started

During the spring and summer of 1981, my son and his friends would meet at our house on Saturday mornings to explore the wonders of TI Logo, the first commercial Logo package. Two of the boys were "forced" into dragging their young siblings along. Although this was a chore for the boys, it provided the spark that ignited my interest in kids, computers, and Logo.

While the older boys were developing games, videos, and other more exotic procedures, I started the young children off by letting them explore what the turtle could do. Some weeks later I got curious. I asked the younger children to put their pencil in the middle of a piece of blank paper and go forward 100 turtle steps, right 153 turtle turns, and forward 100 again. On other pieces of paper, I asked them to turn other random angles: 221, 47, 305, for example.

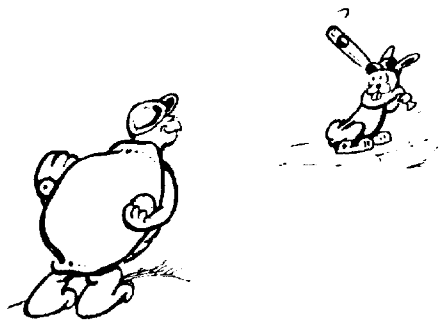
Later, I asked three design engineers to sketch a similar set of random angles. Both they and I were amazed to see that all four of those young children, ages 6, 8, and 10, were able to more accurately visualize random angles than were trained designers. The designers acknowledged they had little to no practice in visualizing random angles like that. They relied on readily available tools.

How did this happen?

To get the young children started in Logo while the older boys and I focused on the "more important stuff," I used an erasable marker to sketch some flowers and rocks on the computer screen. After giving the children some practice with the basic turtle commands, we made up a game.

Who could help the turtle find the flowers without bumping into a rock in the least number of moves? We could just as easily have made this into a game of golf. Who could find the first hole in the fewest stokes?

Finding the flowers led to a more challenging game. When the youngsters had a bit more experience with angles and distances, we started the Turtle Baseball League. This was a very popular group activity when we later began to visit Logo classes and computer clubs.

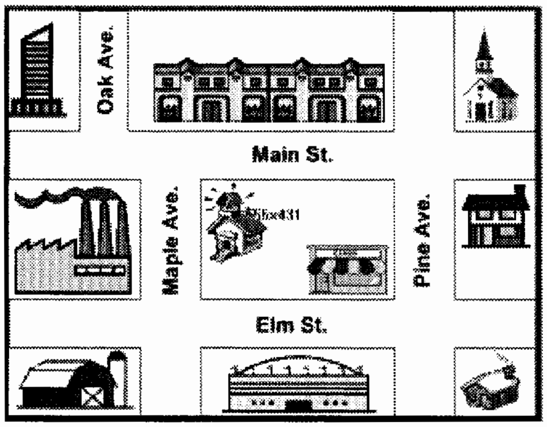


We drew four bases on the screen. Each player had three chances—three strikes—to get to first base or they were out. Then the next player on their team was "up."

Players who got to first base in less than three strikes could use their remaining strikes to continue around the bases. At first, the bases were equidistant from one another. When this game was mastered, random distances and angles were added.

Exploring "Turtle Town"

Taping transparencies to the computer screen was another way of taking the drudgery out of developing visualization skills. Exploring Turtle Town was among the more popular exercises, largely because it offered so many options for different kinds of learning fun.



The first tasks were simple exercises with directions and distances.

1. Start from your home in the lower right-hand corner and find the farm.
2. Now find the factory.
3. Now it's time to go to school.

Then came the challenges. Who could complete a trip in the least number of moves? For example:

1. Go to your friend's house.
2. Take lunch to your friend's father who works at the factory.
3. Go see your cousin who lives in the apartments.
4. Go get some candy at the candy store.
5. Go back home.

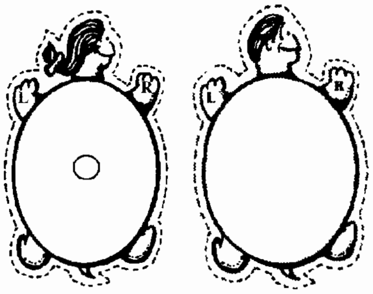
Turtle geography

One nice thing about transparencies is that you can make paper copies. Children have a chance to explore Logo off the computer using a pencil or "turtle" marker. This is also a way of introducing the subject of turtle geography through the maps of Turtle Town. Forward, back, left, and right now take on new meanings: north, south, east, and west. For example:

1. Go north on Pine Avenue to Elm Street.
2. Go west on Elm Street to Maple Avenue.
3. Go north on Maple Avenue to Main Street.
4. What direction must you turn to find the skyscraper at the corner of Main and Oak Streets?

One of the transitions that young children invariably have trouble with is visualizing turtle commands when the turtle is heading south, toward the bottom of the screen.

Making turtle pencils has been a big help. Paste turtles like the ones shown below on a piece of cardboard.

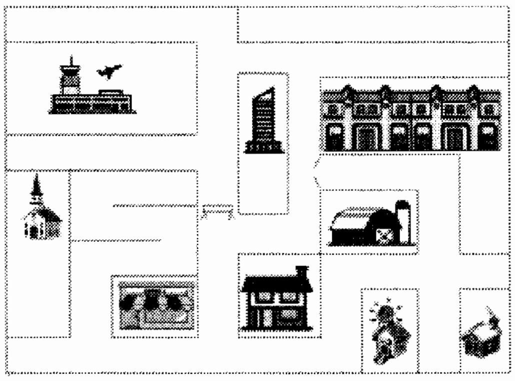


Then push the turtle onto a pencil, leaving enough room so that the child's hand can fit underneath. The trick is to have the pencil mimic the turtle.

Go forward 100. (The turtle moves toward the top of the paper.)
Right 90. (The turtle faces the right edge of the paper.)
Forward 100 (The turtle moves toward the right edge of the paper.)
Right 90 (The turtle faces the bottom of the paper.)

The L and R on the turtle's front feet help keep directions straight.

You can take the concept of Turtle Town maps just as far as you want. The second map shows you a few ideas.



- Add some variables to the map, such as temporary construction barriers.
- Turn the map into a maze.

- Have the children create their own Turtle Town game.

Game design

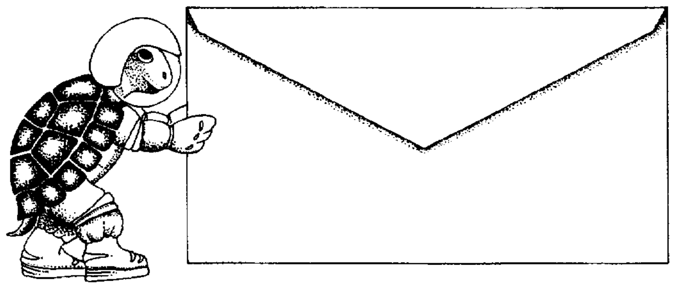
Divide the class into several teams. Each team must then design its own Turtle Town game. When completed, teams play the games developed by each other and vote on which is the best game. This activity can also be shared with other classes in the same or another school.

Game design can be very useful. Not only does it force teams to think through a series of Logo commands, but it also forces them to think logically and strategically. The same goes for players. They must be equally as creative to win. For example:

- One team added the feature that any player who drove off the road had to go to jail and miss three turns.
- Members of another team developed a game-winning strategy: They turned the turtle into a helicopter so they could get across town directly, rather than move through the streets and make all those turns. (All they did was lift the pen up.)

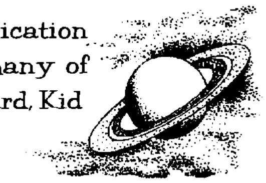
Working with maps and developing games lead to all sorts of wonderful possibilities. Hopefully the ideas offered in this article will lead to many more of your own. If you are working with older children, why not take a look at the "Logo Geography" article in this issue. It offers methods for bringing the subjects of maps, map making, geography, and turtle geometry together. ▲

Jim Muller has had a lifelong interest in translating various technologies into understandable and persuasive programs. In 1981, Muller and his son organized the first Logo users group, the Young Peoples' Logo Association, which eventually grew into a worldwide 6,000 member organization. In 1985, the YPLA merged with CompuServe, where it became The Logo Forum. Today, Muller is a computer training and marketing consultant in the Dallas/ Fort Worth metroplex. Look for Jim's ideas and activities in the newly updated *The Turtle's Discovery Book*, an expanded two-volume set to be published by Harvard Associates, Inc. later this year. You can reach Jim by e-mail at 76703.3005@compuserve.com or on CompuServe at 76703,3005.



Push the envelope with MicroWorlds

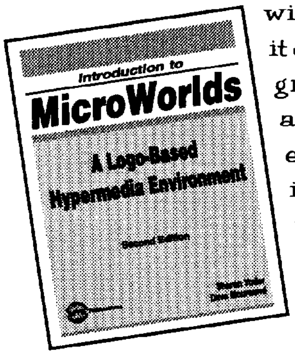
Launch your students toward stimulating new classroom challenges and adventures. They can soar to a galaxy of new skills using MicroWorlds.



MicroWorlds is an exciting application that offers in a single program many of the features available in HyperCard, Kid Pix, and LogoWriter.

You're the pilot. Get ready now to grab the controls, blast off, and explore these exciting new worlds with your class! To help you prepare your crew for this fantastic voyage, Sharon Yoder and Dave Moursund back at mission control have developed a 222-page flight manual titled Introduction to MicroWorlds—A Logo-Based Hypermedia Environment.

As a computer application, MicroWorlds contains a wide range of features that make it easy and fun to work with color graphics, sound, text, and animation. As a programming environment, MicroWorlds includes a powerful and modern version of the Logo programming language.



It's time to broaden your universe. The countdown is underway. Have a nice trip!

International Society for Technology in Education
Customer Service Office
480 Charnelton Street, Eugene, OR 97401-2626 USA
Phone: 800/336-5191 *(US and Canada)*
Phone: 541/302-3777 *(International)*
Fax: 541/302-3778
America Online: ISTE CompuServe: 70014,2117
Internet: cust_svc@ccmail.uoregon.edu
World Wide Web: http://isteonline.uoregon.edu

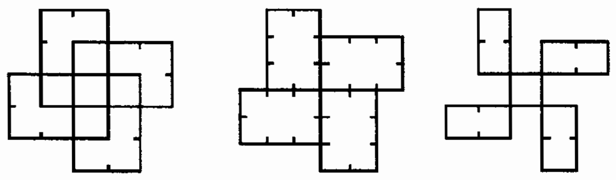
HYPERCARD® IS A REGISTERED TRADEMARK OF CLARIS CORPORATION, SANTA CLARA, CA, U.S.A. KID PIX® IS A REGISTERED TRADEMARK OF BRØDERBUND SOFTWARE, NOVATO, CA, U.S.A. LOGO®, LOGOWRITER® AND MICROWORLDS® ARE REGISTERED TRADEMARKS OF LOGO COMPUTER SYSTEMS, INC., HIGHGATE SPRINGS, VT, U.S.A.

Wumpus, Whimsy, or Gloop?

by Ihor Charischak, President of CLIME (Council for Logo & Technology in Math Education)

Spiros (from *CLIME Microworld*, Vol. 1) continues to be one of my favorite activities in teacher workshops and with students. I've done such a session many times and I find that I learn something new every time.

Spiros draws spirolaterals that can be categorized as Wumpuses, Whimsies, or Gloops.


Wumpus Whimsy Gloop

Experimenting

The main procedure **S** takes three integers that are less than 20 as input. Go ahead—try various inputs to **S**. Look at the designs and collect data. Initially there appears to be no correlation between the numbers and the design. But after some looking and thinking, patterns emerge. See the CopyMe! page on page 10.

Can you figure out what makes a Wumpus a Wumpus, a Whimsy a Whimsy, and a Gloop a Gloop? That is the first step.

After my workshop participants have used the **S** procedure to generate various designs, I ask, "Is there any relationship between the designs (Whimsy, Wumpus, and Gloop) and the numbers? Can you predict what the shape will be from the numbers?"

Conjecturing

Teachers come up with all sorts of conjectures, some of which are fruitful and others less so. For example, one group of three teachers noticed that **S 1 1 1**, **S 2 2 2**, **S 3 3 3**, and so on formed squares that were always Wumpuses. Thus, their initial conjecture was: If all the numbers are the same, you get a Wumpus.

This was correct, but it didn't seem to help them in situations when the numbers are not all the same. So they decided to see what happens when they keep two numbers the same and vary the third (Figure 1). No light bulbs went off immediately. They were all surprised that **S 2 2 5** was a Gloop. I walked away from the group and then returned a few minutes later, noticing that one of them was exploring Gloops. She eventually came up with a pattern that she generalized to all three categories.

Arriving at a solution

She noticed that **S 2 2 5** produced a Gloop that had a 1 x 1 square in the middle. She then looked at the other Gloops (see the Copy Me!

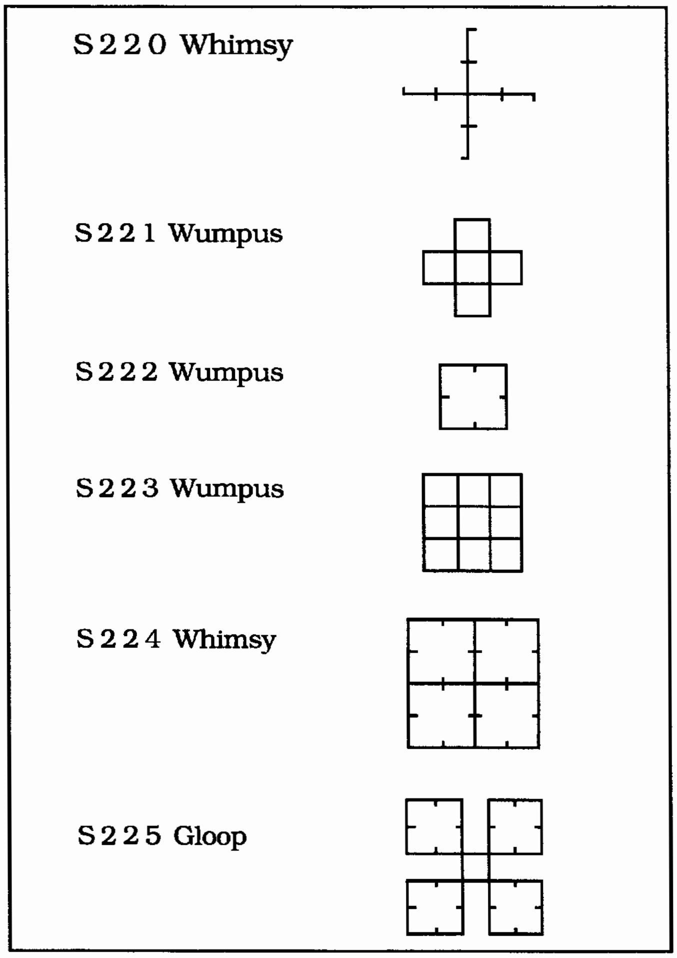
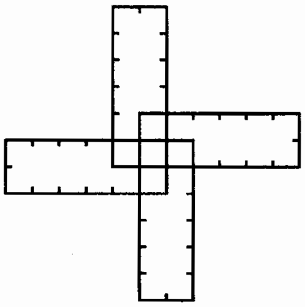
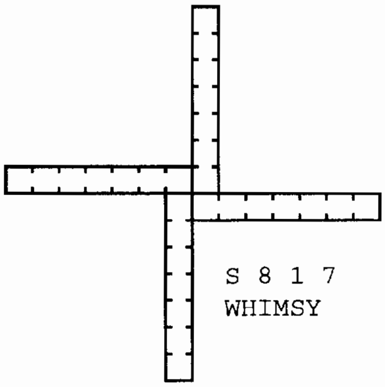


Figure 1

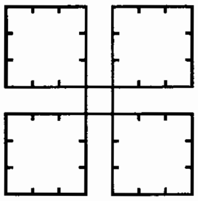
Spiros Data



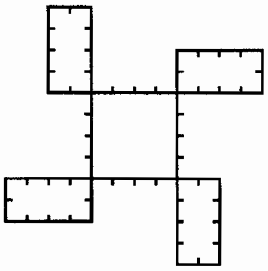
S 2 7 6
WUMPUS



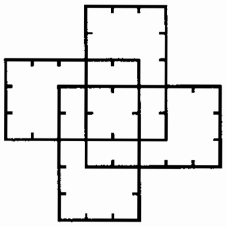
S 8 1 7
WHIMSY



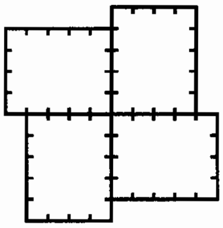
S 3 3 7
GLOOP



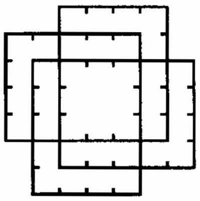
S 10 4 2
GLOOP



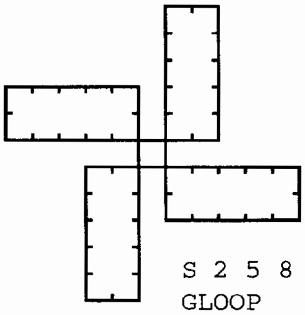
S 3 5 6
WUMPUS



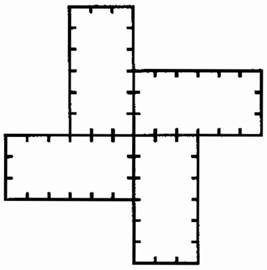
S 4 5 9
WHIMSY



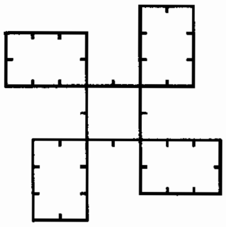
S 6 4 5
WUMPUS



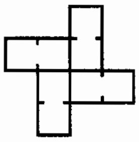
S 2 5 8
GLOOP



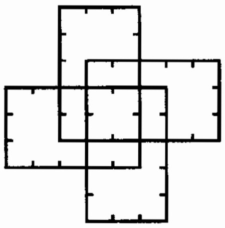
S 3 9 6
WHIMSY



S 3 7 2
GLOOP



S 1 2 3
WHIMSY



S 5 3 6
WUMPUS

page) and recorded the information in a table. It wasn't long before she discovered a pattern for Gloops that she was able to extend to Whimsies and Wumpuses.

See if you can discover how she did it! Think about it before looking at her conjectures on page 34.

The program

This program was written using Terrapin's Logo PLUS for the Macintosh. It can be easily translated to other versions of Logo.

After entering all the procedures, type **startup** to begin.

```
to startup
cleartext
print []
print [SPIROS]
print []
print [Type S followed by three single digit
  numbers. These numbers represent the
  distances of three forward motions.]
print []
print [A right turn will be made after each
  forward motion. This pattern will repeat
  until the path closes.]
print []
end

to s :first :second :third
error.trap scale
spirolateral :first :second :third
end

to spirolateral :first :second :third
draw hideturtle
penup setx -60 pendown
repeat 4 [fwd :first right 90 fwd :second
  right 90 fwd :third right 90]
report.type
end

to fwd :distance
repeat :distance [forward :scale tick.mark]
end

to tick.mark
right 90 forward 2 back 2 left 90
end

to scale
if (or :first > 16 :second > 16  :third >
  16) [make "scale 4 stop]
if (or :first > 8 :second > 8    :third > 8)
  [make "scale 8 stop]
if (or :first > 4 :second > 4    :third > 4)
  [make "scale 10 stop]
make "scale 12
end

to error.trap
if :first > 19 [warning :first toplevel]
if :second > 19 [warning :second toplevel]
if :third > 19 [warning :third toplevel]
end

to warning :number
print sentence :number [is too large an
  input, please reenter with all inputs less
  than 20.]
end

to report.type
sort.inputs
if :s + :m > :l [print [This is a WUMPUS.]]
if :s + :m = :l [print [This is a WHIMSY.]]
if :s + :m < :l [print [This is a GLOOP.]]
end

to sort.inputs
if :first > :second [make "l :first make
  "others [second third]] [make "l :second
  make "others [first third]]
if :third > :l [make "l :third make "others
  [first second]]
if thing first :others > thing last :others
  [make "m thing first :others make "s thing
  last :others] [make "m thing last :others
  make "s thing first :others]
end
```

▲

In addition to founding CLIME in 1987 and supporting it ever since, ***Ihor Charischak*** is program manager for the Center for Improved Engineering and Science Education (CIESE) at Stevens Institute of Technology. His latest project is managing a 3-year NSF-funded mentorship project that involves 40 math teachers (Grades 7–10) from 16 school districts in New Jersey who have been learning to use technology in their classrooms and to share their knowledge with their colleagues. For more information about CLIME you can write to Ihor at 10 Bogert Avenue, White Plains, NY 10606 or send e-mail to icharisc@stevens-tech.edu.

The Mystery of the Moving Billiard Ball

by Robert Macdonald

The idea of creating a statistical and data-collecting microworld dealing with the path of a billiard ball was the result of my coming across a group of activities in a book by Harold R. Jacobs (1970).

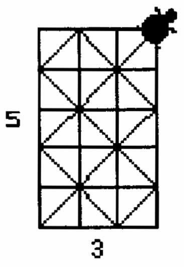
The ordinary billiard table is about half as wide as it is long, some 5 by 10 feet. Jacobs suggests stylizing the game to produce a delightful series of puzzles, for which he merely intimates solutions.

Students should have some prerequisite skills before undertaking this microworld. First, they should have sufficient skills with Logo software, in this case, LogoWriter. They should be able to move in and out of programs, give commands with or without inputs, add labels, and print material with relative ease. In addition, students should have had enough experience in collecting data and utilizing that data to make sensible predictions and arrive at well thought-out conclusions. It is essential that students have some skill in reducing ratios to their lowest terms. I have explored this microworld with fourth-grade students.

The setup

Suppose we are to set up a billiard table in the form of a grid made up of squares. The only ball is struck from one corner so that it always moves at a 45-degree angle to the sides of the table. As it strikes a wall, the ball is always deflected from its path at a 90-degree angle until it exits on arrival at one of the corners of the table.

For example, let's take a table with a length of five units and a width of three units. Suppose we enter at a 45-degree angle at the lower left corner. The ball will move forward on a diagonal through three squares before hitting the wall and deflecting to the left at a 45-degree angle to the wall (but at a right angle to its own path). It then moves forward through two more squares until it is again deflected left at a 45-degree angle to the wall. Following the path of the billiard ball in Example 1, we note that it strikes the wall four more times prior to exiting at the upper right corner.



Example 1

The program

To provide a computer experience in constructing a billiard table and playing on it, a program is essential. The following procedures, written in LogoWriter for the Macintosh, have proven effective.

```
to grid :length :rows :columns
clearpage
place
repeat :rows [row :length :columns go.back]
seth 0
st
hypotenuse :length
move
end

to clearpage
if not front? [flip]
rg ht ct cc
end

to place
pu setpos [-192 77] pd
end

to row :length :columns
repeat :columns [square :length forward
  :length]
end

to square :length
seth 90
repeat 4 [forward :length right 90]
end
```

```
to go.back
seth 270
forward :length * :columns
seth 180
forward :length
end
to hypotenuse :length
make "hypotenuse sqrt (2 * (:length *
  :length))
end

to move
move.ball readchar
end

to move.ball :letter
if equal? :letter "c [right 45]
if equal? :letter "f [forward :hypotenuse]
if equal? :letter "b [back :hypotenuse]
if equal? :letter "r [right 90]
if equal? :letter "l [left 90]
if equal? :letter "u [pu]
if equal? :letter "d [pd]
if equal? :letter "e [pe]
if equal? :letter "s [stop]
move.ball readchar
end
```

Analyzing the program

The main procedure **grid** draws squares representing a billiard table upon the screen. **Grid** requires three inputs: (a) the length of each square in turtle steps, (b) the number of rows (length) desired in the table, and (c) the number of columns (width) in the table.

For our purposes, 15 turtle steps provides a good size for the side of each square. For practical purposes, a 12 x 12 grid would most likely be the largest grid desired by most users, although students very likely will want to experiment with larger sizes. A setup subprocedure places the turtle before the grid is drawn.

Because a billiard ball (the turtle) will be traveling on the diagonal (hypotenuse) of a square, we need a procedure to determine the length of that hypotenuse. In the procedure **grid**, the hypotenuse takes the input of the length of the square.

Now we must move the turtle (billiard ball) around the grid (billiard table). Keys can be programmed to move the ball on command. This is accomplished by using the procedure **move**.

The ball will always enter from the lower left corner. The key C sets the turtle in the lower left corner of the grid at a 45-degree heading, which permits it to move on the hypotenuse produced by the diagonal cutting through each square. The 45-degree heading is produced by the command **seth 45**. The key F moves the turtle forward the distance of one hypotenuse. Key B moves the turtle backward the distance of one hypotenuse. Key R produces a right rotation of 90 degrees. Key L produces a left rotation of 90 degrees.

Because errors may have to be corrected, four additional keys are helpful: Key U creates the command Pen Up; Key D produces Pen Down; Key E produces Pen Erase; and Key S, Stop, provides an exit from the **move** procedure. If you forget to provide that key, after you have drawn your grid and traced the action of the ball, you will not be able to get out of the program and move forward to do the necessary labeling.

After creating a table, it is possible to label it. This labelling might include dimensions, exits, and the number of bounces the ball takes off a side (the wall of the billiard table).

Providing an instructional sheet for students might initially prove helpful. Be sure to give a sample **grid** instruction:

```
grid 15 10 4
```

15 = the side of each square in turtle steps
10 = the number of rows
4 = the number of columns

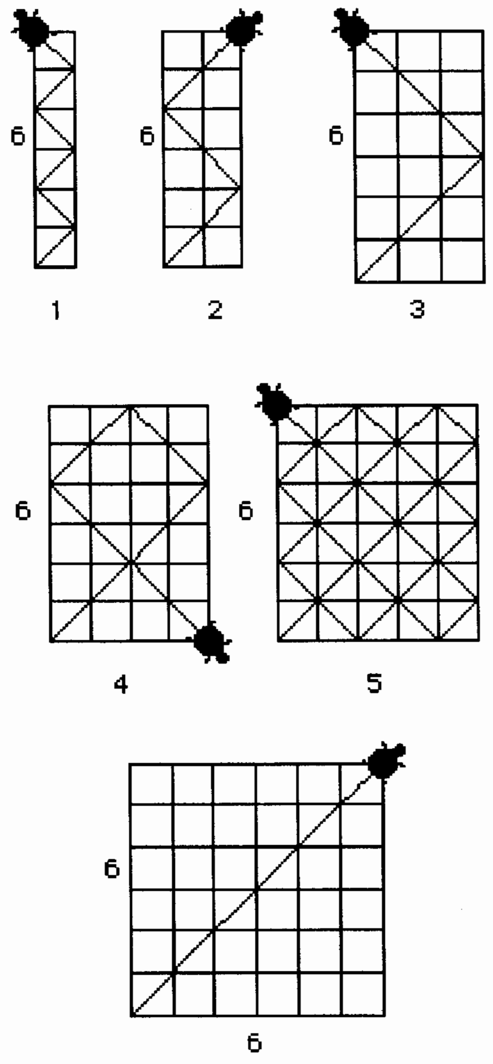
In addition, provide an interpretation for the keys available in the **move** procedure.

C =	start (changes the turtle's heading from 0 to 45, equivalent to **right 45**)		
F =	Forward :hypotenuse		
B =	Back :hypotenuse		
R =	Right 90	**D** =	Pen down
L =	Left 90	**U** =	Pen up
S =	Stop	**E** =	Pen erase

You may need to review how to label your grid. In LogoWriter for the Macintosh, merely click the mouse on the work area and use the cursor to position your work. On an Apple IIe or IIGS you must know how to use Label mode and Turtle Move mode.

Preliminary observations

If we were to draw grids indiscriminately, the paths of the ball would appear to be markedly unpredictable. A slight change in dimension may cause a wildly unexpected path. It is apparent that the shape of the table foreshadows the path of the ball in ways that are still not clear. But what determines the shape of the table? The length and the width. If we change these dimensions, we introduce the concept of variables. Thus the path that the ball will take is predisposed by the shape of the table, which is fixed by its dimensions.

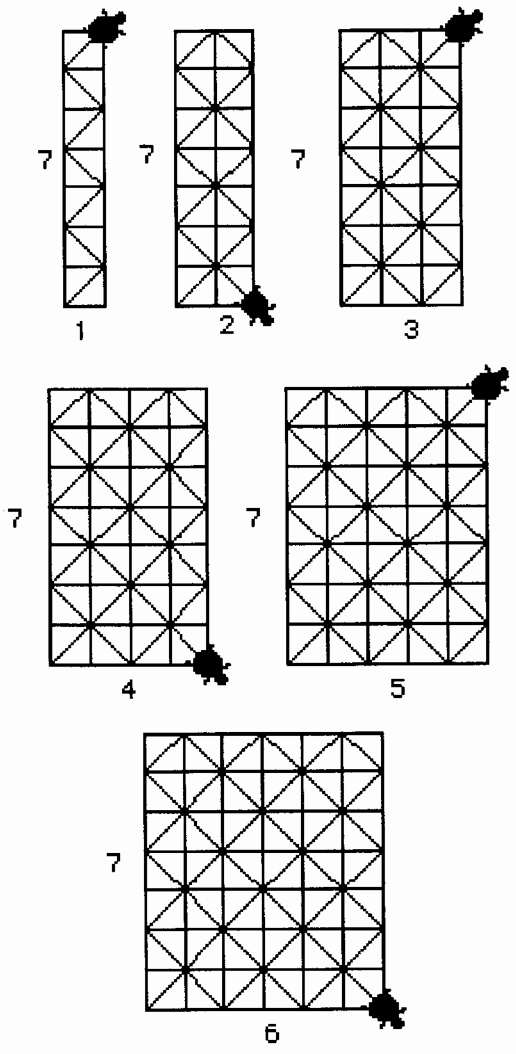


Example 2

It would make our investigation far simpler if we had just one variable. Thus, let us keep the length constant, but vary the width.

If we were to draw a set of six billiard tables with a length of six units, but vary the widths by 1, 2, 3, 4, 5, and 6 units, the results prove interesting. What dimensions (length by width) produce the simplest path? The most complicated path? What can we determine about the number of bounces off the walls? (See Example 2.)

If we were then to draw a set of seven tables with a length of seven units but vary the widths by 1, 2, 3, 4, 5, 6, and 7 units, the results differ markedly from the preceding set. In many tables the turtle traverses every square. Is it possible to deduce which tables those will be? (See Example 3.)



Example 3

These billiard grids provoke all sorts of questions that relate to grids of varying dimensions. Will the ball (turtle) always exit at a corner, or will it rebound from the walls eternally? Will the ball ever return to the original point of origin? If the ball does exit at a corner, and you know the length and width of the grid, will it be possible to predict the exit point and the number of bounces to ensure that exit? Will it be possible to predict the total number of right and left bounces that the ball will make during its journey?

Methodology

Hands-on activities with graph paper and computer-related exercises producing billiard tables may provide ample statistical evidence with regard to exits, number and types of bounces, and patterns of journeys. Drawing conclusions through observation and noting regularities and irregularities in patterning create a wonderful microworld for inductive reasoning.

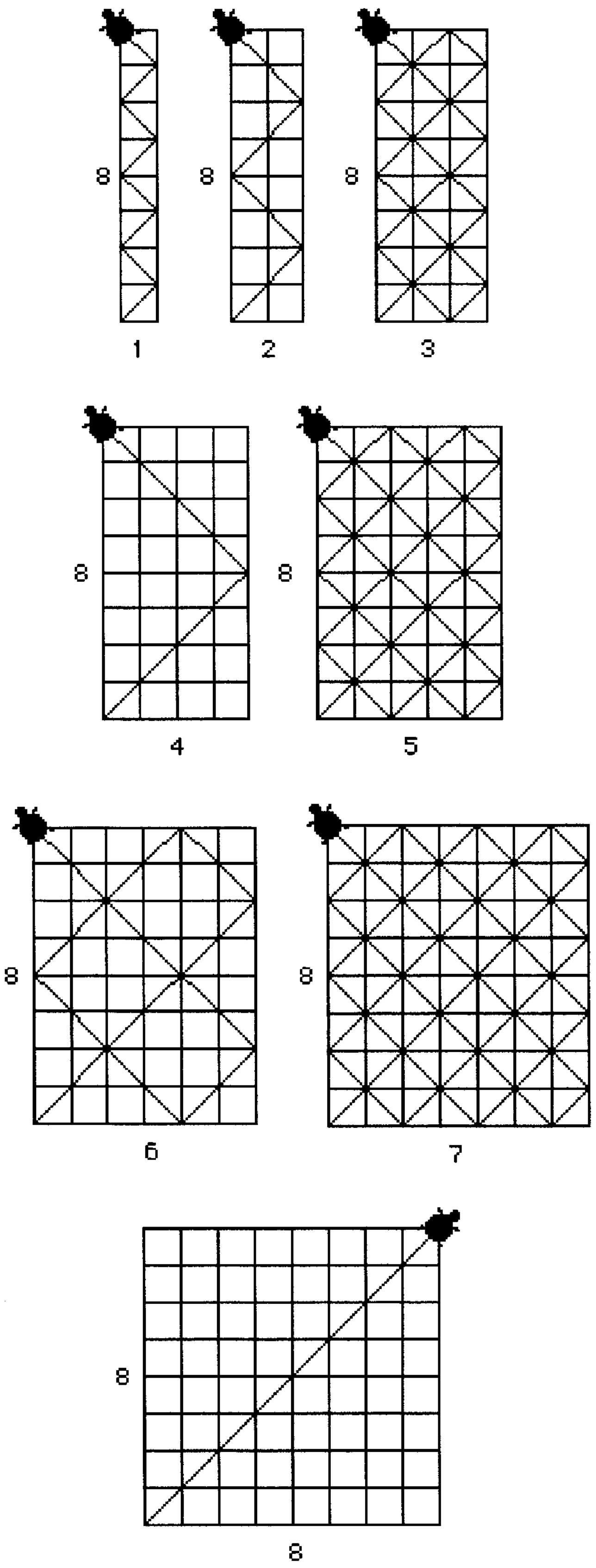
Although inductive reasoning is important in clarifying mathematical ideas, there is a basic weakness. If we draw our evidence from a limited base, we run the risk of drawing conclusions from limited data. Thus, we must be certain that we have a wide enough spread to reduce the risk of predicting possibilities from a faulty set of data.

For example, we could deduce that if we were to draw a set of tables with a length of 8 (8 x 1, 8 x 2, 8 x 3, etc.), the ball would always exit in the upper left corner. This is true up through 8 x 7. However, 8 x 8 produces a marked change, hence the need for a wider spread of data. (See Example 4.)

The concept of the hypotenuse

The turtle's path on our billiard table is along the diagonal of each square. This path, the hypotenuse, is the side of a right triangle that is opposite the right angle.

Students should be given some explanation about the hypotenuse and how it relates to the length of the side of the square. The most direct approach might be to draw a large square with its diagonal. Simple measurement of a side with a comparison to the measurement of the diago-



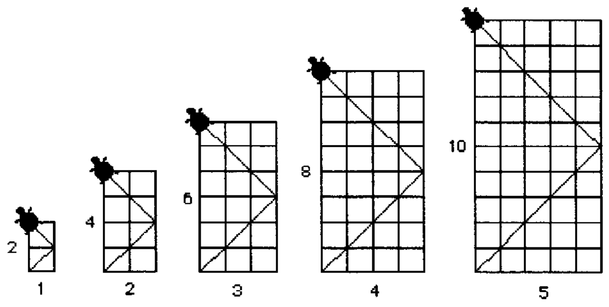
Example 4

nal should prove the need for a formula to compute the distance of the hypotenuse. More able students may examine the computer procedure to discover the formula. Most students will be satisfied with simply measuring distances.

The concept of the ratio

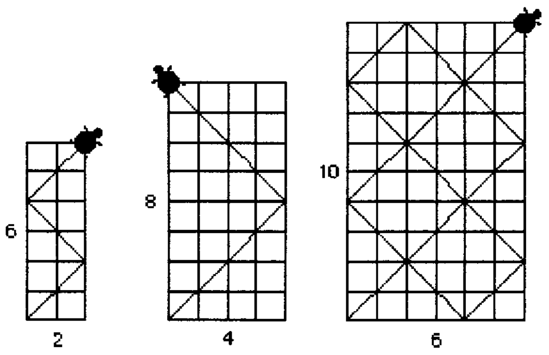
A ratio is a fraction. It is a way of making comparisons between two numbers. A reduction of these numbers by division, producing a simpler ratio, may open up interesting relationships. To discover the simplest ratio between two numbers, divide both by their largest common factor. For example: 6/3 = 2/1.

In this microworld, simplifying ratios is easily demonstrated through tables. Students quickly discover that the path of a ball depends upon the shape of the table. If the shapes of tables are the same, paths should be identical. Discover this through graphing. In Example 5, the ratios are identical. 2/1 = 4/2 = 6/3 = 8/4 = 10/5; thus, all paths are identical.



Example 5

Are the paths in Example 6 similar? No. Why not? The simplest ratios are not equal. 6/2 = 3/1, 8/4 = 2/1, 10/6 = 5/3. The ratio 3/1 is not equal to either 2/1 or 5/3.



Example 6

The study of the path of a billiard ball on a table deals with mathematics, but is not entirely devoted to calculations. Simple addition and the reduction of ratios are just basic requirements. Yet with these skills, students can do surprising things.

Student participation

Each student should have ample opportunity to gather and record statistical information on a full gamut of billiard tables. Generally, a series of lengths from 1 through 10 with widths from 1 through 12 to 15 should represent a broad enough sampling to provide sufficient evidence for making predictions and arriving at sensible conclusions.

Students can record data for each trial in the following columns:

- dimensions of the table
- sum of the dimensions
- number of bounces
- exit corner (UL, UR, LR)

In addition, each child should keep a personal account of graphing and predictions. Booklets of computer printouts of billiard tables and large wall charts of each series should be posted for class use, along with laws that are postulated.

The bounce

The class initially viewed bounces in two ways. Some students preferred to count only a hit off a wall as a bounce. Others considered an entry, a hit off a wall, and an exit as valid bounces. We settled on the latter.

Definition of a bounce:

1. Count the entry (lower left corner).
2. Count the exit.
3. Count each hit off a side.

Law for the number of bounces

After defining the bounce, students can investigate how many bounces varying tables can produce. It may be easier to have the investigation begin with a constant length of one and vary the width, for example, 1 x 1,1 x 2, 1 x 3, 1 x 4, and so on.

Children will soon discover that adding dimensions will total the number of bounces for each table. However, if we introduce the series of threes (3 x 1, 3 x 2, 3 x 3, 3 x 4, etc.), irregularities appear. For example, a table of 3 x 3 does not have six bounces, but rather two. A 3 x 6 table does not have nine bounces, but three. If children have had sufficient experience in reducing ratios, they soon are aware that a reduction in ratios solves the problem. Reinforcement by continuing to project solutions to odd-numbered lengths should be encouraged.

When the discovery of reducing the ratios of dimensions has been assimilated, the investigation of even-numbered lengths is much easier to handle.

Law of the number of bounces:

1. Reduce the dimensions to the lowest ratio.
2. Add the dimensions together.

By this time students should be able to create projection sheets and determine with some accuracy the number of bounces for a complete series of billiard tables. They should be encouraged to check their predictions by graphing solutions. You may wish to put each series on a different color of paper to make organizing data much easier.

Laws of exits

Students will have produced collectively a body of statistical data and visual information to permit the discovery of the laws of exits.

Again, it may be easier to start with the 1x series. All of these tables exit either to the upper right (UR) or lower right (LR). Projection sheets are now essential. Have the students compile them and prove them by graphing.

In the 2x series we discover exits to the upper left (UL), upper right (UR), and lower right (LR). None exit where the turtle enters.

In the 3x series we find that all exits are to the UR and LR. By this time some students may deduce that the same will be true of all odd length series (5x, 7x, 9x, etc.). It is the even length series that produces variety. Why? Experiments will prove that exits for 4x, 6x, and 10x are to the UL, UR, LR, but never to the lower left (LL). The 8x series will exit only to the UL and UR. There no exits at the point of entry nor at the LR.

It will be apparent to many children that the reduction of dimensions to lowest ratios and whether these relationships may be considered odd or even numbers play an important part in bringing order out of seeming chaos.

Through observation a class may determine these laws of exiting:

Law 1: If table length is an odd number and the width is even, the turtle exits to the Lower Right (LR).

Law 2: If table length and width are odd numbers, the turtle exits to the Upper Right (UR).

Law 3: If table length is even and the width is odd, the turtle exits to the Upper Left (UL).

Law 4: If both table length and width are even, reduce the ratio of the dimensions to the lowest terms and follow Law 1, 2, or 3.

This may be shown concisely as follows:

Laws of Exits

	Length	Width	Exit
Law 1	odd	odd	UR
Law 2	odd	even	LR
Law 3	even	odd	UL
Law 4	even	even	

Reduce Ratio to Lowest Terms

	Length	Width	Exit
Law 1	odd	odd	UR
Law 2	odd	even	LR
Law 3	even	odd	UL

Patterning

Eventually students should be drawn to examining patterns. In their study of ratios they have discovered that similar patterns may be masked by unreduced ratios. In addition, some students may be confused by the rotation of a pattern. This posed little difficulty for many of my classes because the students had had much experience examining tangrams and pentominoes.

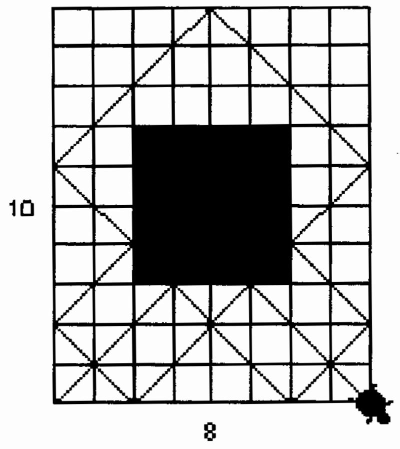
Some classes developed pattern sheets to record similar patterns. The students discovered that the following ratios were pertinent to the study: 1/1, 1/2, (2/1), 2/3, (3/2), 3/4, (4/3) 3/5, (5/3). Students were quick to recognize patterns that were simply extensions of dimensions.

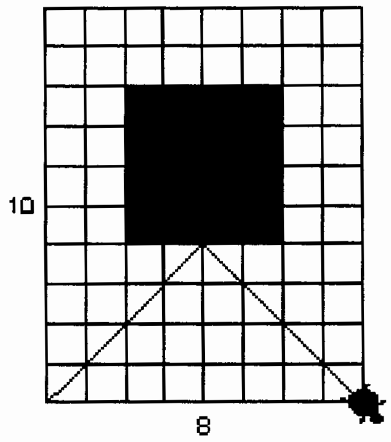
Further explorations

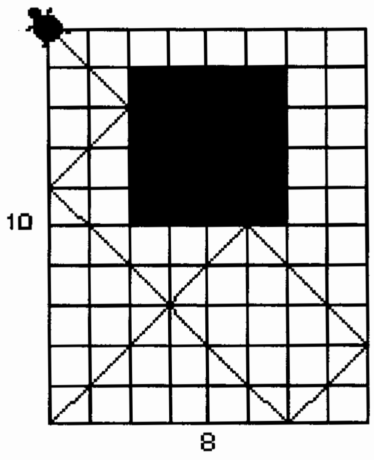
What if an entry point other than the lower left were introduced? What if exit points other than corners were permitted? What if we blocked off part of the playing area of a billiard table? Could we predict the number of right and/or left bounces if we knew the dimensions of a table?

Students soon concluded that entry at a different corner would have no impact on their statistical data. Hence this part of the project was quickly discarded, although a few students made forays into this area.

Allowing exit points in addition to corners opened up interesting vistas. It was suggested that exits at the midpoints along the lengths were traditional. Those who play these table games had a particular interest in spending some of their spare time graphing examples. However, none were able to do much work beyond preliminary graphing.







Example 7

More students were struck by the possibilities offered by billiard tables with an area blocked off within the playing field. (See Example 7.) In a 10 x 8 grid, we block off a 4 x 4 area. In the first example, the symmetrical patterning is marked. In the second sample, the symmetry is expressed simply. The path of the ball in the third sample no longer possesses symmetry. These examples merely hint at possibilities. This aspect is worthy of further study.

Predicting whether bounces will turn right or left may be one of the more fascinating aspects of this study. However, we can only suggest possibilities. This part of the study may not be suitable for fourth-grade investigation. The preceding portions occupied a large amount of time. Predicting the direction of bounces off the sides of walls would appear to require a much greater expenditure of time. Hence, we eliminated this investigation from the microworld.

The following commentary briefly summarizes what someone may be able to attempt. Remember that students decided that a bounce counts an entry, a hit off a wall, and an exit. In this summary we will exclude the entry and the exit in considering bounces.

The laws of right and left bounces:

1. Simplify the ratios to lowest terms.
2. If the length minus the width equals one, the bounces off the wall are always to the left.
3. If the length minus the width equals two, the bounces off the wall go as many times to the left as to the right. Mathematically, we may add the sides, subtract two (this excludes the entry and exit) and divide by two. Half the bounces will go to the right, the other half to the left.
4. If the length minus the width equals three, add the length and width. Subtract two. If this number of bounces off the wall is divisible evenly by three, then one-third of the bounces move to the left, one-third to the right, then one-third to the left. If not divisible evenly by three, one-third of the bounces move left, while one-third plus two (the remainder) hit to the right, then return one-third to the left.
5. If the length minus the width equals four, then we see four changes of direction. If the bounces off the wall are divisible evenly by four, then the bounces will alternate by quarters in direction. If not divisible evenly by four, then the second and third quarter change of direction will have two more bounces each than the first and fourth quarters.
6. It appears that Law 5 may be applicable to greater differences in length and width of billiard tables. ▲

Robert Macdonald
Hawthorne Meadows
10225 Nancy's Blvd.
Grosse Ile, MI 48138-2128

Reference

Jacobs, H. R. (1970). *Mathematics, a human endeavor: A book for those who think they don't like the subject.* San Francisco: W. H. Freeman and Co.

Logo in Motion: Exploring Elementary Physics

by Nancy Flynn

What does recursion have to do with Newton's first law of motion? Everything!

When I found myself in a physics class last winter, I couldn't help but notice that certain phenomena could not be demonstrated in the classroom. For instance, take Newton's first law of motion: A body at rest will remain at rest or, if in motion, will remain in motion in a straight line unless interfered with by some external influence. This external influence in the physics lab is friction. Therefore, the principle of inertia cannot be demonstrated in the lab, unless of course, you use Logo.

It is very simple to demonstrate Newton's first law of motion using recursion and very simple Logo commands. This procedure starts the turtle in straight-line motion and keeps it in constant motion without the effects of friction. The only way to stop the turtle is to stop the procedure.

```
to motion
forward 1
motion
end
```

Circular motion and acceleration

Recursion can also be used to demonstrate circular motion and acceleration. Whereas in the procedure **motion** there is no change in the speed or direction of the turtle, we can add the element of velocity by changing the turtle's direction in circular motion. We can change the turtle's speed in accelerated motion.

Circular motion:

```
to circular.motion
forward 1
right 1
circular.motion
end
```

The procedure **circular.motion** tracks the turtle moving at a constant speed, but not in a constant direction. The turtle changes direction as it goes around in a circle, creating an effect known as centripetal acceleration; hence the element of velocity is introduced. This procedure can also be used to demonstrate centripetal force; as you watch the turtle, you can see that it always turns to the right, towards the center of the circle.

Accelerated motion:

```
to accelerate
make "move 0
end

to accelerated.motion
make "move :move + 1
forward :move
accelerated.motion
end
```

The procedure **accelerated.motion** uses recursion to demonstrate a change in velocity caused by a change in the speed of the turtle. The turtle starts out at a slow pace and appears to accelerate until it reaches a peak speed.

If you are using LogoWriter, you may be confused by an unusual effect. After the turtle reaches its peak speed, it seems to change speed, but this is only because of the way in which the image of the turtle is displayed on the screen. The apparent slowing down or even backwards motion is because the image of the turtle flashes on the screen after every forward motion. If the turtle moves 10000 and then 10001, the second position may be lower on the screen than the first because of the size of the window; it isn't one pixel higher, as you might expect.

This effect is an optical illusion, similar to one you can see when a bicycle wheel turns. At some point, the spokes appear to go backwards even though the direction of the spin hasn't changed.

To prove this in Logo, turn the turtle **right 33** before starting the procedure **accelerated.motion**. You won't see the same effect as you do when the heading is 0 degrees.

Sound waves

Another situation in which Logo can be a useful tool in physics involves pictorially representing sound waves. After developing procedures that direct the turtle to trace out an approximation of a sine wave (Figure 1), it is then possible to change the shape and size of the wave by changing the relevant commands.

By varying the shape and size of the wave, it is possible to change the size of the wavelength (Figure 2), the amplitude of the wave (Figure 3) and the frequency (Figure 4). It is also possible to experiment with the waves out of phase, as shown in Figure 5.

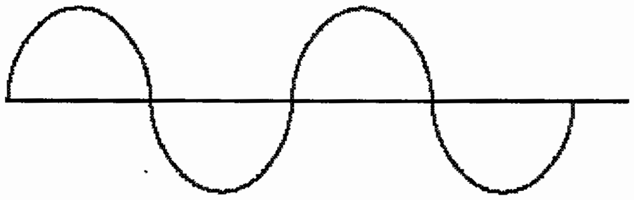


Figure 1. Sine wave

```
to sine.wave
pu setpos [-100 0] pd seth 90 fd 340
pu setpos [-100 0] pd seth 0
crest
trough
crest
trough
end
```

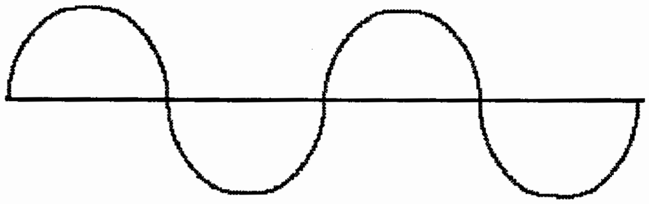


Figure 2. Larger wavelength

```
to larger.wavelength
pu setpos [-120 0] pd seth 90 fd 350
pu setpos [-120 0] pd seth 0
crest1
trough1
crest1
trough1
end
```

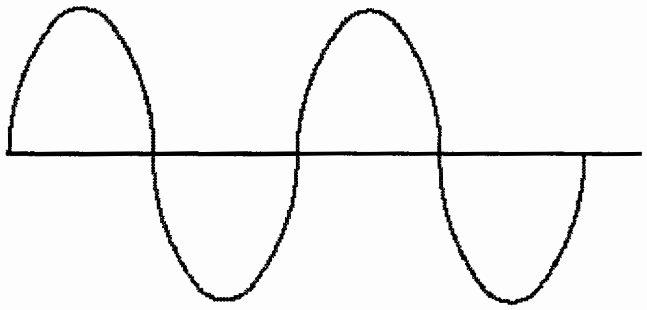


Figure 3. Greater amplitude

```
to greater.amplitude
pu setpos [-100 0] pd seth 90 fd 340
pu setpos [-100 0] pd seth 0
crest2
trough2
crest2
trough2
end
```

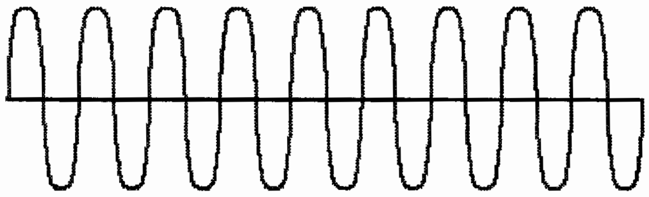


Figure 4. Greater frequency

```
to greater.frequency
pu setpos [-200 0] pd seth 90 fd 350
pu setpos [-200 0] pd seth 0
repeat 9 [curve]
end
```

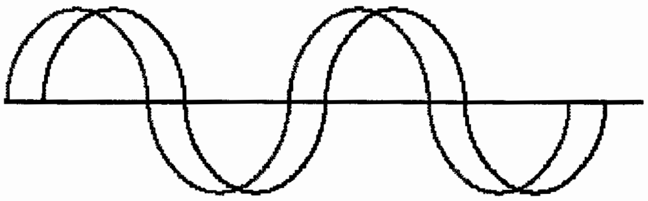


Figure 5. Sine waves out of phase

```
to sine.wave.out.of.phase
pu setpos [-120 0] pd seth 90 fd 350
pu setpos [-120 0] pd seth 0
crest
trough
crest
trough
pu setpos [-100 0] pd seth 0
crest
trough
crest
trough
end
```

Sound wave subprocedures:

```
to crest
repeat 50 [fd 1 rt 1]
repeat 40 [fd 1 rt 2]
repeat 50 [fd 1 rt 1]
end

to trough
repeat 50 [fd 1 lt 1]
repeat 40 [fd 1 lt 2]
repeat 50 [fd 1 lt 1]
end

to crest1
repeat 50 [fd 1 rt 1]
repeat 20 [fd 1 rt 2]
fd 10 rt 1
repeat 20 [fd 1 rt 2]
repeat 50 [fd 1 rt 1]
end

to trough1
repeat 50 [fd 1 lt 1]
repeat 20 [fd 1 lt 2]
fd 10 lt 1
repeat 20 [fd 1 lt 2]
repeat 50 [fd 1 lt 1]
end

to crest2
repeat 76 [fd 1 rt .5]
repeat 35 [fd 1 rt 3]
repeat 76 [fd 1 rt .5]
end

to trough2
repeat 76 [fd 1 lt .5]
repeat 35 [fd 1 lt 3]
repeat 76 [fd 1 lt .5]
end

to curve
repeat 45 [fd 1 rt .2]
repeat 18 [fd 1 rt 9]
repeat 45 [fd 1 rt .2]
repeat 45 [fd 1 lt .2]
repeat 18 [fd 1 lt 9]
repeat 45 [fd 1 lt .2]
end
```

Conclusion

Certain concepts in physics can be particularly difficult to learn because they cannot be demonstrated "in the real world." Logo is a tool that can help to envision these concepts and to clarify principles and laws in physics that may otherwise be difficult to demonstrate. These procedures also reinforce the concept of recursion by applying it to a real-life situation. ▲

Nancy Flynn is a computer specialist in the St. Paul Public Schools in Minnesota. She is currently writing a book on multicultural projects using LogoWriter and how they can be integrated into the classroom curriculum. She can be reached by e-mail at nkflynn@aol.com.

References

Asimov, I. (1966). *Understanding physics motion: Sound and heat*. London: George Allen & Unwin Ltd.

Hewitt, P. G. (1993). *Conceptual physics*. New York: Harper Collins College Publishers.

Osborne, J. (1987). New Technology and Newtonian Physics. *Physics Education, 22*(6), 360–364.

Exploring Turtle Geography

by Jim Muller

A number of years ago, while giving a presentation at an educational conference, a school administrator asked me why anyone should learn Logo. After all, there were no jobs to be had as Logo programmers.

In my view, learning to program using Logo is like learning to drive using a bulldozer. Neither choice is very practical. The bulldozer is not very practical as transportation, but for some tasks, it is invaluable. Learning Logo may not be very practical as a programming language. However, learning *with* Logo can be an invaluable experience.

Using Logo to explore the many sides of mathematics is what most people think of first. However, Logo is also an invaluable aid for exploring language arts, not to mention other languages. It was fun figuring out what procedure titles written in Dutch, German, French, and Japanese meant by what they did.

Geography came up when we began to explore the question of "wrapping."

Some young people in an elementary school computer club got curious about what happened when the turtle went off one edge of the screen and then showed up on the opposite edge. Using the transparency of Turtle Town taped to the screen, as described in the "Exploring the Logo Screen" article in this issue, the question hit them when they watched the turtle go off the screen and then walk down the street from the other side again.

Did it travel behind the screen?

Exploring maps

Maps provided the answer. It gave us the chance to look at various types of maps to see what happened when you flattened a three-dimensional object into two dimensions. It was easy enough to picture flattening the world into a two-dimensional map; if they traveled off one edge of the map, they would automatically "wrap" to the same spot on the opposite edge.

Exploring Turtle Town grew into the game of "Where in the World is...?"

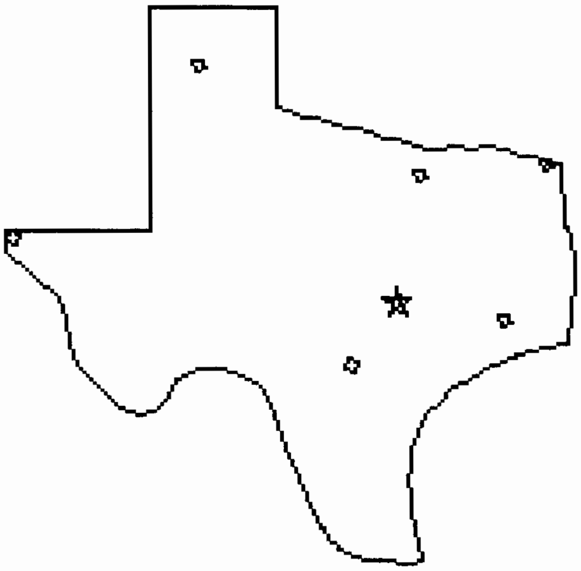
Teams of students taped transparencies of the United States on the screen and then tried to stump other teams by laying out the most difficult route for the turtle to follow. The transparencies were small and did not show much detail. Therefore, teams had to refer to other maps to find out where cities were located.

This introduced the whole idea of scale. On the screen, the turtle moved **forward 50**, for example. But when this represented moving across the map of the United States, how many miles did 50 turtle steps cover? Another question was that if there were 50 miles to the inch on one map, how could the team measure that distance on another map where there were 500 miles to the inch?

Soon the questions arose, "Why don't we make our own map?"

Texas, here we come!

The map of Texas includes the cities of Amarillo, Austin, Dallas, El Paso, Houston, San Antonio, and Texarkana. And, because game design was a very important activity with this group, they made a game out of it.



It took some discussion but the group finally figured out that the actual miles on the screen map were the number of turtle steps times 0.187. For example:

550 miles * 0.187 = 103 turtle steps

Once they had this figured out, it was easy enough to use a larger map to create the map of Texas on the screen with the cities accurately placed on that map. Then came the fun of designing the game.

The game offered the students good practice guessing directions and drawing mileage to scale. After working with Texas, the group did several other states. Each time, they kept adding new features.

1. Take a look at the Texas program and see if you can develop the same type of procedure for your state.
2. Add other landmarks from your state—lakes, rivers, state parks, and so on.
3. When a player moves from one place to another correctly, ask a random question about that place. Think up five or six questions for each location you put on your map. Make sure that the player answers correctly.

Can you think of other ways to explore Logo geography?

The program

This program was written using PC Logo. It can easily be adapted for other versions of Logo using the guidelines below. To receive an electronic copy of this program, send a request via e-mail to: 71760.366@compuserve.com. Type **begin** to start.

For your version of Logo, you may need to:

- change **if *condition* [*action*]** to: **if *condition* then *action***
- change **random 7** to **1 + random 7**
- change **setbg** to **bg**
- abbreviate commands (**ct** for **cleartext**, etc.)
- add the tool procedures listed at the end

```
to begin
cleartext
pr [Logy and Morf are going to take a trip
  through the Lone Star State.]
wait 80
pr [You're going to help them find their way
  around.]
wait 80 pr []
pr [The trip starts at] first.city
wait 80
cs ht
texas
start
pr [Your destination is] second.city
pr [Press any key to continue.]
ignore rc
game
end

to first.city
make "city random 7
if :city = 1 [pr [amarillo]]
if :city = 2 [pr [dallas/ft. worth]]
if :city = 3 [pr [texarkana]]
if :city = 4 [pr [houston]]
if :city = 5 [pr [austin]]
if :city = 6 [pr [san antonio]]
if :city = 7 [pr [el paso]]
end

to second.city
make "dest random 7
if :dest = :city [second.city stop]
if :dest = 1 [pr [amarillo]]
if :dest = 2 [pr [dallas/ft. worth]]
if :dest = 3 [pr [texarkana]]
if :dest = 4 [pr [houston]]
if :dest = 5 [pr [austin]]
if :dest = 6 [pr [san antonio]]
if :dest = 7 [pr [el paso]]
end

to game
pr [Logy, what direction do we turn?]
pr [How many miles do we have to go?]
pr [Can you give Logy a hand?]
pr [What direction do they turn?]
make "dir readword
pr [How far?]
make "turns readword
if :dir = "lt [left :turns]
if :dir = "rt [right :turns]
pr [How many miles do they have to travel?]
make "far readword
pu forward :far * .187
check
end
```

```
to check
if :dest = 1 [make "x1 -30 make "y1 81]
if :dest = 2 [make "x1  33 make "y1 45]
if :dest = 3 [make "x1  66 make "y1 53]
if :dest = 4 [make "x1  57 make "y1  9]
if :dest = 5 [make "x1  25 make "y1 12]
if :dest = 6 [make "x1  12 make "y1 -3]
if :dest = 7 [make "x1 -84 make "y1 32]
if and xcor < ( :x1 + 5 ) xcor > ( :x1 - 5 )
  [cheers]
wait 50
redo
end

to cheers
if and (ycor < ( :y1 - 5 )) (ycor > ( :y1 +
  5 )) [try.again]
repeat 3 [setbg 3 flash setbg 13 flash]
setbg 15 wait 5
play2
end

to flash
wait 20
pr [WOW! You're quite an explorer.]
wait 50
end

to try.again
pr [You silly tourist! Can't you read a
  map?]
wait 15
start
game
end

to play2
make "city :dest
start
pr [Your next destination is] second.city
pr [Ready to go? Press any key.]
ignore rc
game
end

to start
pu
if :city = 1 [setpos [-30 81]]
if :city = 5 [setpos [ 25 12]]
if :city = 2 [setpos [ 33 50]]
if :city = 7 [setpos [-84 32]]
if :city = 4 [setpos [ 57  9]]
if :city = 6 [setpos [ 12 -3]]
if :city = 7 [setpos [ 66 53]]
st seth 0 pd
end
```

```
to texas
ht pu setpos [-84 36] pd
seth 90 fd 41 lt 90
fd 63 rt 90 fd 36 rt 90 fd 28
red.river
louisiana
gulf
riogrande
cities
end

to red.river
repeat 4 [seth 110 fd 9 lt 20 fd 3]
repeat 2 [seth 75 fd 5 rt 30 fd 5]
repeat 2 [seth 110 fd 5 lt 20 fd 3]
end

to louisiana
rt 85 fd 17 lt 20
repeat 17 [fd 1 rt 2]
seth 185 fd 18
end

to gulf
seth 265
repeat 32 [fd 1 lt 1]
repeat 3 [coast lt 4]
repeat 3 [rt 10 fd 4 lt 15 fd 5]
end

to coast
rt 15 fd 5 lt 30 fd 5
end

to riogrande
rt 90 fd 4 seth 275 fd 10
repeat 20 [fd 1 rt 3]
repeat 5 [fd 1 lt 2]
rt 15 fd 12 lt 17 fd 4
rt 20 fd 12 lt 10 fd 8
repeat 2 [lt 20 fd 5]
repeat 18 [fd 1 lt 4] lt 15 fd 3
repeat 18 [fd 1 rt 6]
fd 15 rt 20 fd 3 rt 15
fd 15 lt 8 fd 3 lt 30
fd 20 seth 350 setpos [-84 36]
end

to cities
amarillo
dfw
texarkana
houston
austin
san.antonio
el.paso
end

to city
repeat 4 [fd 3 rt 90]
end
```

```
to amarillo
pu setpos [-30 81] pd
city
end

to dfw
pu setpos [33 50] pd
city
end

to texarkana
pu setpos [69 53] pd
city
end

to houston
pu setpos [57 9] pd
city
end

to austin
pu setpos [25 12] seth 18 pd
repeat 5 [fd 8 rt 144]
end

to san.antonio
pu setpos [12 -3] pd
city
end

to el.paso
pu setpos [-84 33] pd
city
end

to ending
clearscreen cleartext
pr [The Lone Star State thanks you!]
pr [Hurry back, ya' he-yah!]
end
```

Your version of Logo may also need the following tool procedures:

```
to readword
output first readlist
end

to ignore :thing
end
```

for PC Logo, add:

```
to setpos :list
setxy :list
end
```

for Terrapin Logo or Logo PLUS/Apple II, add:

```
to setpos :list
setxy first :list last :list
end
```

▲

Jim Muller is the moderator of the Logo Forum on CompuServe. You can read more about him at the end of his other article in this issue, "Exploring the Logo Screen." You can reach him by e-mail at 76703.3005@compuserve.com or on CompuServe at 76703,3005.

Recursion and the Stack

by Don E. Ryoti

"But, why does it do that?" is the question students ask when they first encounter the results of **kounta 3** and **kountb 3**. The students in this case are college students who have chosen the mathematics emphasis of the K–4 or 5–8 education major.

```
to kounta :numb
if :numb = 0 then print [done] stop
print :numb
kounta (:numb - 1)
end

to kountb :numb
if :numb = 0 then print [done] stop
kountb (:numb - 1)
print :numb
end
```

Kounta produces:

```
3
2
1
done
```

Kountb produces:

```
done
1
2
3
```

The stack

The concept of a stack is important for understanding what the computer is doing. The usual intuitive notion for a stack is a stack of cafeteria trays. The top tray on the stack is the first one removed.

When the computer is given **kountb 3**, it determines that **kountb** is in the workspace and that **kountb** has the input 3. Once the computer has determined that it can execute the procedure with the input, it creates a stack. In this case the stack has 4 lines:

```
if 3 = 0 then print [done] stop
kountb (3 - 1)
print 3
end
```

Note that the 3 and not :numb is used in the stack. Also note that the third line has a 3 and not a 2. Students have a tendency to compute the (3 - 1) in line two and utilize the result in the subsequent lines.

After putting these four lines on the stack, the computer then executes the top line of the stack. Because 3 = 0 is false, the **then** portion is not executed. The line has been executed and is removed from the stack. The machine executes the **kountb (3 - 1)** statement. The machine again determines that **kountb** is a procedure in the workspace and that it is being provided with the input 2. The four lines of **kountb** with input 2 are placed on top of the stack replacing the line **kountb (3 - 1)**. The stack now is:

```
if 2 = 0 then print [done] stop
kountb (2 - 1)
print 2
end
print 3
end
```

In presentations to students, it is important to pause and reflect upon how the stack started with four lines, that two lines were executed, and that there is a new level of four lines that is above two of the original lines. The computer executes the top line of a stack. "But, nothing will ever be printed" is a common reaction. Let's continue.

The top line of the stack is executed. Because 2 = 0 is false, **kountb (2 - 1)** is executed. The four statements of **kountb** with input 1 are placed on the stack, which is now:

```
if 1 = 0 then print [done] stop
kountb (1 - 1)
print 1
end
print 2
end
print 3
end
```

The top line of the stack is executed. Because 1 = 0 is false, **kountb (1 - 1)** is executed. The four statements of **kountb** with input 0 are placed on the stack, which is now:

```
if 0 = 0 then print [done] stop
kountb (0 - 1)
print 0
end
print 1
end
print 2
end
print 3
end
```

The top line of the stack is executed. Now, 0 = 0 is true, a new development. The **then** portion is executed and **done** is printed on the screen. **Stop** tells the computer to cease the execution of the current procedure, that is, to ignore the lines on the stack through the end of the current procedure. In this particular example, the computer ignores **kountb (0 - 1)** and **print 0**.

The top statement on the stack is **print 1** and the machine prints 1; the **end** statement marks the end of that particular level. The top statement on the stack is now **print 2** and the machine prints 2; the **end** statement marks the end of that particular level. The top statement of the stack is now **print 3** and the machine prints 3; the **end** statement marks the end of that particular level. The machine determines that there is nothing on the stack and it returns to what is called "toplevel."

Including the **end** statement in the stack is useful in developing the stack concept with students. The **ends** mark the levels of the stack.

Ways of looking at the stack

In the classroom, the instructor can write the statements and draw lines through them as they are executed or skipped. Students can follow what is happening, but when the process is completed and the students look at the crossed-out lines, some are confused because the sequencing is not shown.

It is helpful to many students to create their own stack using 3 x 5 index cards or pieces of paper with each statement on a separate card. Have the entire procedure **to kountb :numb** on a sheet of paper representing the workspace. Talk through creating the stack and executing the lines as presented previously.

How else can the creation and execution of the stack be presented? Numbering lines to show the sequence is one possibility. In the representation of the stack for **kountb 3** below:

1. The levels are numbered.
2. The numbers at the left represent an ordering of the lines when they are entered on the stack
3. The numbers at the right represent the ordering of the lines when they are executed.

It might be desirable to have another set of numbers to represent the order of when the **print** lines are executed.

```
level 4:
13  if 0 = 0 then pr [done] stop    7
14  kountb (0 - 1)
15  print 0
16  end                             8

level 3:
 9  if 1 = 0 then pr [done] stop    5
10  kountb (1 - 1)                  6
11  print 1                         9
12  end                            10

level 2:
 5  if 2 = 0 then pr [done] stop    3
 6  kountb (2 - 1)                  4
 7  print 2                        11
 8  end                            12

level 1:
 1  if 3  = 0 then pr [done] stop   1
 2  kountb (3 - 1)                  2
 3  print 3                        13
 4  end                            14
```

Exploring recursion

A basic recursive procedure will have three components:

- a conditional statement for ending the recursion
- an action
- the recursive call

Three statements can be ordered in six different ways. What does each of the six different arrangements produce?

Write the stack for each of the procedures with input 3. Some students are certain that when the recursive call is first, the machine will continue building the stack forever. They forget that machines have a finite quantity of memory. Each procedure is listed with its stack in the Appendix at the end of this article.

Note that one difficult error to spot is that the student may have the procedure **kountd** call **kountc** rather than **kountd**.

If you want to introduce **first** and **butfirst**, students can investigate the six ways to arrange the three lines for the basic recursive procedure that prints the first item of the list and removes the first item from the list. Students will discover that when the recursive call is first, the machine stops not because of "too many recursive calls" or "no more stack space," but because **butfirst** does not like the empty list as input.

```
to tella :stuff
tella (butfirst :stuff)
if :stuff = [] then pr [done] stop
print (first :stuff)
end
```

Recursion is a valuable tool in Logo; it is also an important concept both in computer science and mathematics. Understanding how the computer works with recursive procedures is useful for Logophiles and prospective teachers.

Appendix

Your version of Logo may require the following **if** construction:

```
if :numb = 0 [pr [done] stop]
```

Procedure Kounta:

```
to kounta :numb
if :numb = 0 then pr [done] stop
print :numb
kounta (:numb - 1)
end
```

The stack for **kounta 3**:

```
level 4:
13  if 0 = 0 then pr [done] stop   10
14  print 0
15  kounta (0 - 1)
16  end                            11
level 3:
 9  if 1 = 0 then pr [done] stop    7
10  print 1                         8
11  kounta (1 - 1)                  9
12  end                            12
level 2:
 5  if 2 = 0 then pr [done] stop    4
 6  print 2                         5
 7  kounta (2 - 1)                  6
 8  end                            13
level 1:
 1  if 3  = 0 then pr [done] stop   1
 2  print 3                         2
 3  kounta (3 - 1)                  3
 4  end                            14
```

Note that **kounta 3** prints 3, 2, 1, done on four different lines.

Procedure Kountc:

```
to kountc :numb
print :numb
if :numb = 0 then pr [done] stop
kountc (:numb - 1)
end
```

The stack for **kountc 3**:

```
level 4:
13  print 0                        10
14  if 0 = 0 then pr [done] stop   11
15  kountc (0 - 1)
16  end                            12
level 3:
 9  print 1                         7
10  if 1 = 0 then pr [done] stop    8
11  kountc (1 - 1)                  9
12  end                            13
level 2:
 5  print 2                         4
 6  if 2 = 0 then pr [done] stop    5
 7  kountc (2 - 1)                  6
 8  end                            14
level 1:
 1  print 3                         1
 2  if 3  = 0 then pr [done] stop   2
 3  kountc (3 - 1)                  3
 4  end                            15
```

Note that **kountc 3** prints 3, 2, 1, 0, done on five different lines.

Procedure Kountd:

```
to kountd :numb
print :numb
kountd (:numb - 1)
if :numb = 0 then pr [done] stop
end
```

The stack for **kountd 3**:

```
level 6:
17  print -2                        11
    etc., with additional levels
level 5:
13  print -1                         9
14  kountd (-1 - 1)                 10
15  if -1 = 0 then pr [done] stop
16  end
level 4:
13  print 0                          7
14  kountd (0 - 1)                   8
15  if 0 = 0 then pr [done] stop
16  end
level 3:
 9  print 1                          5
10  kountd (1 - 1)                   6
11  if 1 = 0 then pr [done] stop
12  end
level 2:
 5  print 2                          3
 6  kountd (2 - 1)                   4
 7  if 2 = 0 then pr [done] stop
 8  end
level 1:
 1  print 3                          1
 2  kountd (3 - 1)                   2
 3  if 3  = 0 then pr [done] stop
 4  end
```

Kountd 3 prints 3, 2, 1, 0, -1, and so on, but at some point stops with a message that there is no more stack space or that there are too many recursive calls.

Procedure Kounte:

```
to kounte :numb
kounte (:numb - 1)
print :numb
if :numb = 0 then pr [done] stop
end
```

The stack for **kounte 3**:

```
level 4:
13  kounte (0 - 1)                   4
    etc., with additional levels
level 3:
 9  kounte (1 - 1)                   3
10  print 1
11  if 1 = 0 then pr [done] stop
12  end
level 2:
 5  kounte (2 - 1)                   2
 6  print 2
 7  if 2 = 0 then pr [done] stop
 8  end
level 1:
 1  kounte (3 - 1)                   1
 2  print 3
 3  if 3 = 0 then pr [done] stop
 4  end
```

Kounte 3 prints no numbers and at some point stops with a message that there is no more stack space or that there are too many recursive calls.

Procedure Kountf:

```
to kountf :numb
kountf (:numb - 1)
print :numb
if :numb = 0 then pr [done] stop
end
```

The stack for **kountf 3**:

```
level 4:
13  kountf (0 - 1)                   4
    etc.,  with additional levels
level 3:
 9  kountf (1 - 1)                   3
10  if 1 = 0 then pr [done] stop
11  print 1
12  end
level 2:
 5  kountf (2 - 1)                   2
 6  if 2 = 0 then pr [done] stop
 7  print 2
 8  end
level 1:
 1  kountf (3 - 1)                   1
 2  if 3 = 0 then pr [done] stop
 3  print 3
 4  end
```

Kountf 3 prints no numbers and at some point stops with a message that there is no more stack space or that there are too many recursive calls.

Don E. Ryoti is a professor of mathematical sciences at Eastern Kentucky University. He has introduced Logo to fourth through eighth graders and regularly teaches a 3-hr. CSC Programming in Logo course, which is a requirement for K–4 and 5–8 mathematics emphasis education majors. He can be reached at Wallace Building 313, Eastern Kentucky University, Richmond, KY 40475-3133; e-mail: matryoti@acs.eku.edu.

Logo: Search and Research
Getting Computers Into the Classroom

by Douglas H. Clements and Julie Sarama

What will it take to get computers well implemented? We Logophiles know that this means a significant change in education. We have also learned that it is a significant challenge to do it well.

The Ministry of Education in Ontario is committed to full-scale, high-quality implementation of computers in all schools. Recognizing the challenge, they generated a report that pulls together what is known about implementing educational change (Fullan, Miles, & Anderson, 1988).

Ten factors that make or break innovation

The researchers organized their findings according to ten critical factors. These factors are: clarity, consensus, quality and practicality, central office commitment, a plan for implementation, professional development and assistance, monitoring and problem solving, principal's leadership, community support, and environmental stability. Some of these might seem "obvious"; however, understanding them and designing effective interventions is enormously challenging.

Clarity

Teachers need to be clear about what to do and what to change. Information helps, but true understanding comes only when teachers work with the innovation in their classrooms and talk about what they are doing with others.

Designers of software wish to increase details and specificity in software documentation; however, clarity is not increased by doing so. Instead, we need to help teachers develop and try out effective *teaching practices* in relation to the software. If we don't, teachers often learn only the easiest surface features of software. And unless students come to understand how programs work and their potential, they will not learn much.

Consensus

Teachers need to agree on the need, appropriateness, and priority of the software innovation. Such consensus can develop over time, but only in favorable circumstances. For example, the administration must be supportive, and continuing assistance should be available. There must be a core of teachers who support the innovation. Other teachers are more likely to "sign on" if there is parent pressure, ongoing collaboration with other users in the same setting, and adequate training. The innovation must also be effective. It is interesting that the researchers, who studied a variety of software types, found that Logo encouraged motivated and committed teachers. Traditional curriculum structures and entrenched, less creative software were found to be deleterious to the teachers' implementation of educational change.

Quality and practicality of the change

Chances for successful change are best when at least some of the benefits for students are immediately apparent to teachers: This is key to developing commitment for skeptical teachers. Also important is how "practical" the innovation seems. The innovations seen as practical address perceived needs and include concrete how-to-do-it information.

Central office commitment and support

General endorsement and verbal support from the central office are not enough. Actions are critical. Pressure must be accompanied by assistance. The communication must be, "We are going to do it right and we will help you." Such help should include adequate resources for training, consultants, release time, and materials.

Commitment is recognized by teachers when real effort is made to identify and solve problems that arise during implementation and real recognition is given to those who progress. Priority must be given to the quality, and not the ease, of implementation; otherwise, most innovations will be "downsized" or adapted away.

Plan for implementation

Implementers need a clear plan. It must be conceptually solid, organizationally practical, and politically sensitive. An effective process explicitly takes into account the other nine factors. Teachers and administrators must be involved, together, at all stages.

Professional development and assistance

Substantive change requires inservice and consultative assistance during implementation. Effective inservice provides explanations with demonstrations, opportunity to practice in nonthreatening contexts, and individual feedback or coaching. Also important are release time and *sustained* sharing and problem solving among teachers.

Monitoring and problem solving

Continuous feedback is needed about progress and problems. The right people should be talking about implementation issues on a regular basis.

Principal's leadership

Principals must take an active role in adopting and carrying out changes, rather than just leaving the responsibility to their teachers. In reality, principals are often less enthusiastic and prepared than their teachers. Those using technology for administrative purposes are more likely to support classroom technology.

Community support

Most innovations proceed without much community awareness and involvement. But when there, community support for or against the innovation is often a critical factor, especially because parents often support their children learning *about* computers more than learning *with* computers. Educating the public is wise.

Environmental stability

Finally, success may be affected by changes in the general organizational and social context. For example, there can be devastating effects of frequent or unexpected changes in administration or project leaders. These can be minimized by ensuring that responsibility for implementation, management, and support is spread among various people, and by considering project continuity when transferring administrators. Finally, organizational support for implementation has to be flexible and tenacious enough to deal with unexpected turbulence in the local and broader environmental context.

Requirements for designing an implementation

The researchers developed design requirements for a comprehensive implementation-support strategy. They included local responsiveness, initial acceptance of an uncertain innovation, provision for increasing clarity, learning maximization, high credibility, overload reduction, sustained responsive assistance, expansion of assistance capabilities, support for user interaction, clarity about principal's role, attention to time reallocating, early success, use of existing human resources, accommodation of old and new uses of microcomputers, nonreliance on software perfection, accommodation of a range of user types, continued updated training for new software and hardware innovations, parent involvement, coordination among multiple agencies, and visible progress and impact.

Strategies to ensure quality implementation

Based on these design requirements, the researchers developed the following comprehensive list of strategies for effective educational innovation with technology, modified for our purposes.

Competence development

- Development of Logo orientation workshop
- Development of classroom management workshop
- Development of workshop series on planning and managing implementation
- Development of principals' workshop on administrative uses of microcomputers
- Development of inservice programs and follow-up services

Consultant development

- Design of a process for identifying and legitimating local consultants

- Training of local consultants
- Design of a workshop series on helping and training skills
- Development of a workshop on inservice training design
- Development of a regional support structure for local consultants
- Professional development of regular subject consultants

Stimulation / facilitation of naturalistic efforts
- Grants for pilot implementation
- Design of a system to support cumulative learning from pilot projects
- Design of a documentation system for pilot projects
- Grants for demonstration sites
- Follow-on grants for local expansion

Diffusing and supporting effective practice
- Design of a system for identifying well-implemented, effective local practices
- Directory of approved practices
- Development of contracts between developers/demonstrators and local boards
- Help lines for accessing software developers and consultants

Summary

The researchers note that most implementers vastly underestimate the difficulty of their task. "It is as if teachers needed to learn to read and to understand what a book is at the same time as proceeding to select books, make lesson plans, and carry them out—all the while continuing with their other 'normal' teaching activities."

Implementing technology in education must receive more attention. Success must be local and early. This requires well-planned, intense, relevant, and sustained assistance as well as direct, active leadership. ▲

Douglas H. Clements, professor at the State University of New York at Buffalo, has studied the use of Logo environments in developing children's creative, mathematical, metacognitive, problem-solving, and social abilities. Through a National Science Foundation (NSF) grant, he developed a K–6 elementary geometry curriculum, *Logo Geometry* (published by Silver Burdett & Ginn, 1991). With colleagues, he is working on the above-mentioned NSF research grant and is finishing a second NSF-funded project, *Investigations in Number, Data, and Space*, to develop a full K–5 mathematics curriculum featuring Logo. With Sarama, he is coauthoring new versions of Logo for learning elementary mathematics. One, Turtle Math, is presently available from LCSI.

He can be reached at:
State University of New York at Buffalo
Department of Learning and Instruction
593 Baldy Hall
Buffalo, NY 14260
E-mail: Clements@acsu.buffalo.edu

Julie Sarama, PhD, is an associate professor at Wayne State University in Detroit, Michigan. She has taught secondary mathematics and computer science, gifted math at the middle school level, and mathematics methods courses. She is coauthor of Turtle Math and is currently designing and programming new versions of Logo.

She can be reached at:
Wayne State University
Teacher Education Division
Detroit, MI 48202
E-mail: Sarama@acsu.buffalo.edu

Time to prepare this material was partially provided by a National Science Foundation Research Grant, "An Investigation of the Development of Elementary Children's Geometric Thinking in Computer and Noncomputer Environments" (NSF MDR-8954664). Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Fullan, M. G., Miles, M. B., & Anderson, S. E. (1988). *Strategies for implementing microcomputers in schools: The Ontario case*. Toronto, Ontario: MGS Publications Services.

Wumpuses, Whimsies, and Gloops: Data Analysis and Conjectures

Here are conjectures about the Wumpuses, Whimsies, and Gloops explored in the article on page 9.

Gloops:

Numbers	Size of resulting square
10 4 2	4
2 5 8	1
3 7 2	2

Conjecture about Gloops:

If the sum of the two smaller numbers is less than the third, you have a Gloop with a square in the middle whose side length is the difference between these values.

Whimsies:

Numbers	Size of resulting square
8 1 7	0
1 2 3	0
4 5 9	0
3 9 6	0

Conjecture about Whimsies:

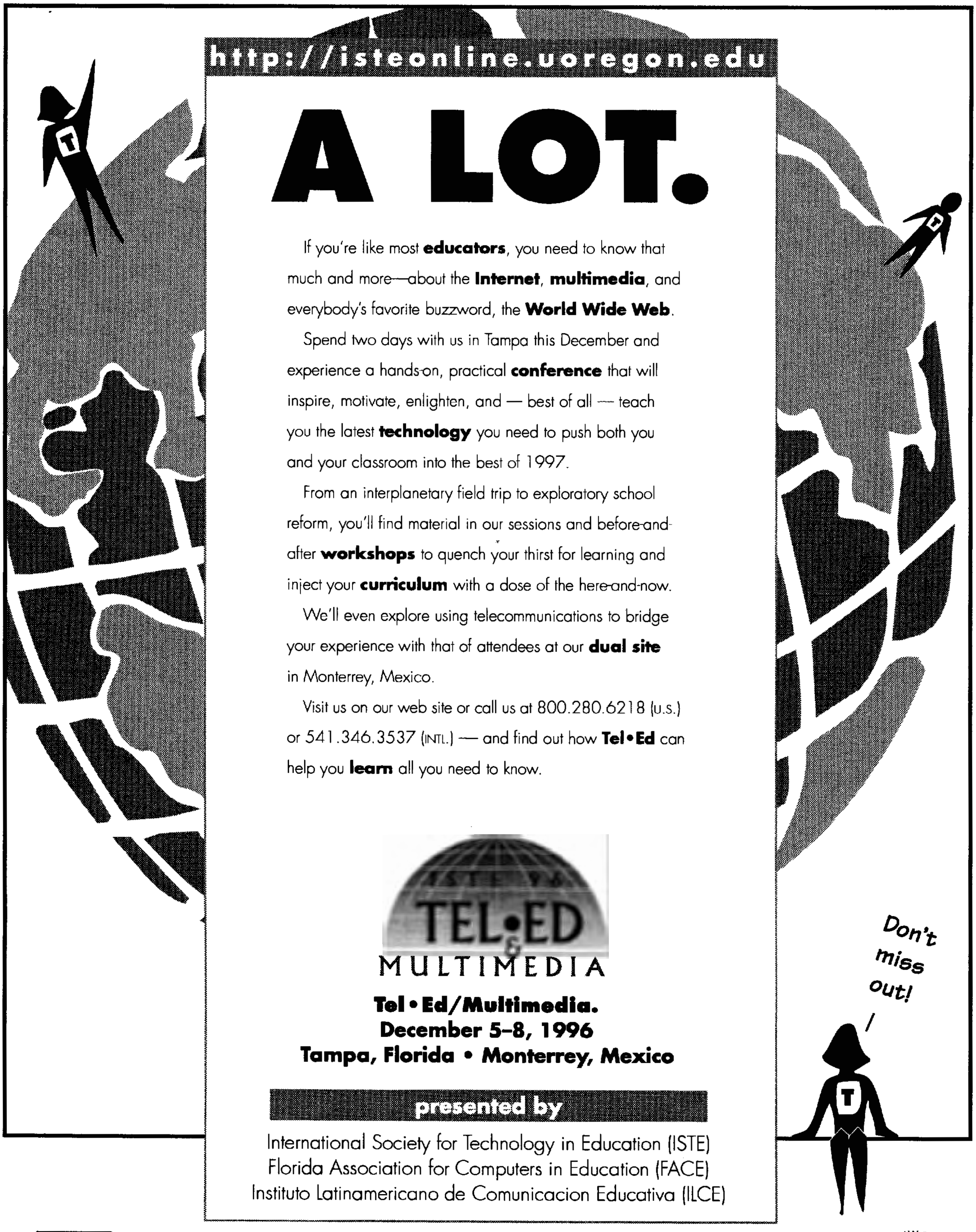
If the sum of the two smaller numbers is equal to the third, you have a Whimsy with no square in the middle.

Wumpuses:

Numbers	Size of resulting square
3 5 6	–2
5 3 6	–2
2 7 6	–1
6 4 5	–3

Conjecture about Wumpuses:

If the sum of the two smaller numbers is greater than the third, you have a Wumpus with a square in the middle created by the overlapping rectangles whose side length is the difference between these values.



http://isteonline.uoregon.edu

A LOT.

If you're like most **educators**, you need to know that much and more—about the **Internet**, **multimedia**, and everybody's favorite buzzword, the **World Wide Web**.

Spend two days with us in Tampa this December and experience a hands-on, practical **conference** that will inspire, motivate, enlighten, and — best of all — teach you the latest **technology** you need to push both you and your classroom into the best of 1997.

From an interplanetary field trip to exploratory school reform, you'll find material in our sessions and before-and-after **workshops** to quench your thirst for learning and inject your **curriculum** with a dose of the here-and-now.

We'll even explore using telecommunications to bridge your experience with that of attendees at our **dual site** in Monterrey, Mexico.

Visit us on our web site or call us at 800.280.6218 (U.S.) or 541.346.3537 (INTL.) — and find out how **Tel•Ed** can help you **learn** all you need to know.

ISTE 96
TEL•ED
&
MULTIMEDIA

Tel • Ed/Multimedia.
December 5–8, 1996
Tampa, Florida • Monterrey, Mexico

presented by

International Society for Technology in Education (ISTE)
Florida Association for Computers in Education (FACE)
Instituto Latinamericano de Comunicacion Educativa (ILCE)

FACE ILCE ISTE INTERNATIONAL SOCIETY FOR TECHNOLOGY IN EDUCATION

ISTE Books & Courseware Order Form

To order ISTE products advertised in this publication, find the product title in the following list and enter it on the form below.
To receive a free Resource Guide with a complete listing of ISTE products and services, please call our toll-free number, 800/336-5191.

Product (• Indicates an ISTE-published title.)	*Member Price*	*Nonmember Price*
• *Introduction to MicroWorlds 2.0—A Logo-Based Hypermedia Environment*	25.15	27.95
• *Introduction to MicroWorlds—A Logo-Based Hypermedia Environment, 2nd Ed.*	25.15	27.95
• *Introduction to Programming in Logo Using Logo PLUS*	13.45	14.95
• *Introduction to Programming in Logo Using LogoWriter, 3rd E.*	24.25	26.95
• *Introduction to Programming Using Terrapin Logo for the Macintosh*	13.45	14.95
• *LogoWriter for Educators—A Problem Solving Approach*	11.70	13.00
• *Logo Learning—Strategies for Assessing Content and Process*	6.25	6.95
• *Logo Musings—Ten Mathematical Encounters Using LogoWriter*	19.75	21.95
• *Math Activities Using LogoWriter (Specify Apple II series, Mac, or MS-DOS data disk)*		
Patterns and Designs	11.65	12.95
Investigations	8.95	9.95
Numbers & Operations	8.95	9.95
High School Math	11.65	12.95
More Investigations	8.95	9.95
More Patterns and Designs	8.95	9.95
Probability and Statistics	8.95	9.95
• *MicroWorlds—Hypermedia Project Development and Logo Scripting*	31.45	34.95
• *Turtle Power—Beginning Graphics With LogoWriter, Rev. Ed. (Specify Apple II series, Mac, or MS-DOS data disk)*	12.55	13.95
• *More Turtle Power—Text, Graphics, and Sound With LogoWriter (Specify Apple II series, Mac, or MS-DOS data disk)*	11.65	12.95
• *Turtle Time Designs—Working With Patterns in LogoWriter (Specify Apple II series, Mac, or MS-DOS data disk)*	8.95	9.95
• *Wacky Words for Traveling Tales—Logo Applications in Language Arts*	8.05	8.95

Receive an additional 18% discount when ordering 10 or more of the same title of ISTE-published products.

Name ____________________ Membership # ____________

School/Business ____________________

Address ____________________

City ____________ State ________ Zip/Postal Code ____________

Country ____________ Phone ____________

Shipping & Handling

$0-$15.99 (subtotal)	add $4.50
$16-$45.99 (subtotal)	$6.00
$46-$75.99 (subtotal)	$7.00
$76-$100.99 (subtotal)	$8.00
$101 or more	8% of subtotal

GST Registration Number 128828431

Code LX1

ORDER

Quantity	Title	Member Unit Price	Nonmember Unit Price	Total Price

PAYMENT OPTIONS

☐ **Payment enclosed.** Make checks payable to ISTE—
International orders must be prepaid with U.S. funds or credit card.
☐ VISA ☐ MasterCard ☐ Discover Card Expiration Date ____________

☐ **Purchase Order enclosed.** Please add $4.00 for order processing—
P.O. not including $4.00 fee will be returned.

☐ **C.O.D.** for U.S. book orders only. You will pay UPS the total upon delivery by check or cash—
ISTE will add $4.75 order processing.

☐ **Airmail.** International orders for Books & Courseware are sent surface mail—
ISTE will bill you the additional shipping charge for airmail.

☐ Send me ISTE membership and subscription information.

☐ Send me a free ISTE catalog.

SUBTOTAL	
Deduct 18% on ISTE-published titles if ordering quantities of 10 or more of the same title	-
SUBTOTAL	
*Shipping and Handling (see box above)	+
*Add Additional 7% of SUBTOTAL if shipped to PO Box, AK, HI, or outside U.S.	+
Add 7% of SUBTOTAL for GST if shipped to Canada	+
If billed with purchase order, add $4.00; If COD, add $4.75	+
TOTAL	=

* If actual shipping cost exceeds this amount, we will bill you for the difference.

ISTE • 480 Charnelton Street, Eugene, OR 97401-2626 USA • Order Desk 800/336-5191 • Fax 541/302-3778
(Effective November 1, 1995, our area code changed to 541)

TURTLE
POWER
LOGO LAUNCHES LEARNING