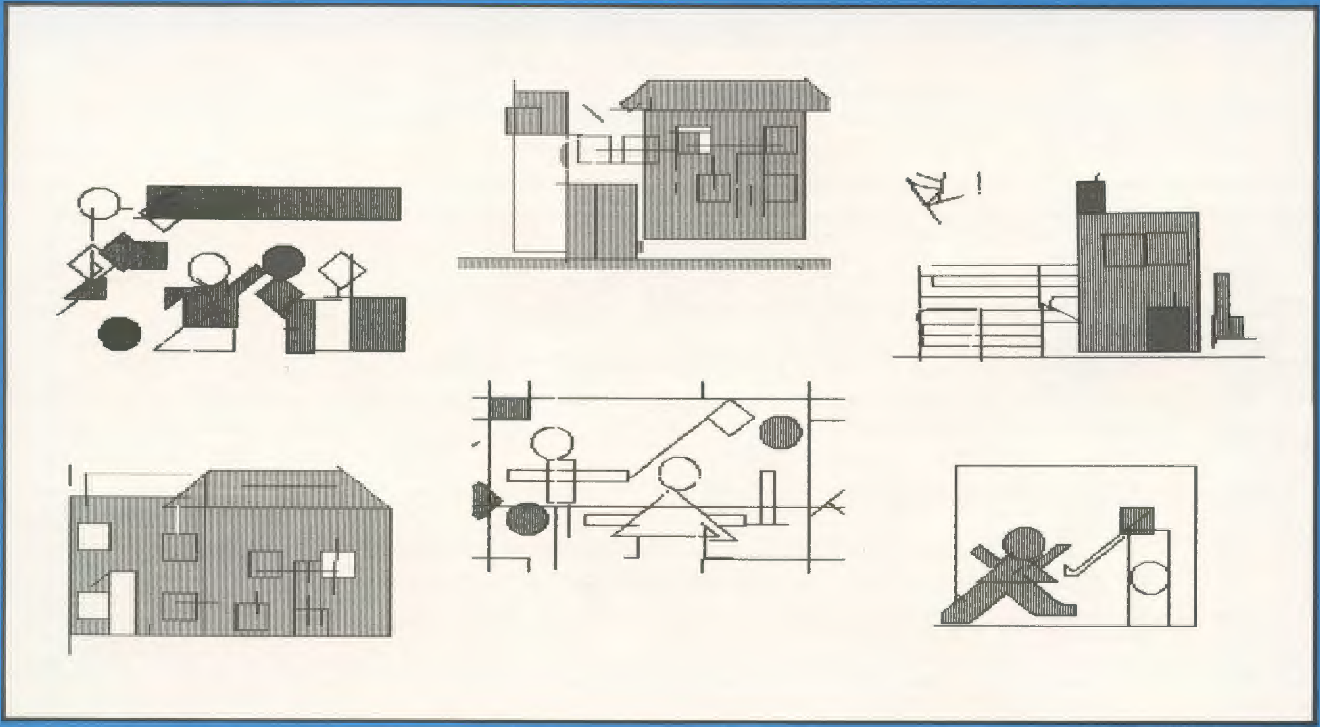
Journal of the ICCE Special Interest Group for Logo-Using Educators

LOGO EXCHANGE

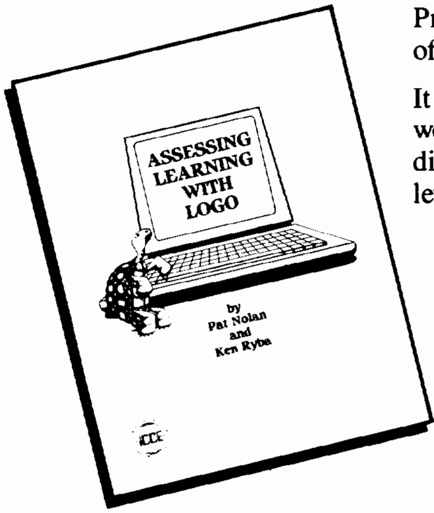
JANUARY 1989 VOLUME 7 NUMBER 5



International Council for Computers in Education

ICCE Publications

Assessing Learning With Logo



Presents the method for *Assessing Learning With Logo* at the levels of basic Turtle commands, repeats and procedures.

It contains all the necessary materials—checklists, assessment worksheets and activities—for developing coding, exploration, prediction, analysis and planning, creativity, and debugging at each level of learning Logo.

The methods and activities have been especially designed to highlight the role of the educator as a "facilitator of learning." In this role educators guide students to reflect on their own thinking as they come into contact with powerful ideas at the beginning levels of Logo.

Single copies are $12.50 plus $2.50 shipping. Call now for a free catalog of ICCE publications.

INTERNATIONAL COUNCIL FOR COMPUTERS IN EDUCATION
ICCE

ICCE, University of Oregon, 1787 Agate St., Eugene, OR 97403. Ph: 503/686-4414.

THE WEST COAST LOGO GROUP PRESENTS

HANDS ON TECHNOLOGY

Friday and Saturday, February 3-4, 1989

The Los Angeles Airport Hilton and Towers, Los Angeles, California

Keynote Speakers: Suzanne Bailey, Seymour Papert, Tom Snyder

panel discussions, individual speaker presentations
35 "hands-on" workshops for Macintosh, IIGS and //e, and IBM
special sessions and hands-on workshops for administrators and coordinators

TOPICS OF SPECIAL INTEREST: HyperCard, desktop publishing, Lego/Logo, LogoWriter, telecommunications, video laserdisc, technology within curriculum frameworks, computers in the arts, problem-solving tools, teaching with simulations, spreadsheet math, technology and literature, school planning for the future, computing for children with special needs, equity and excellence, staff development strategies, model school projects, satellite teleconferencing, instructional television, and video production!

Workshops available on a first-come, first-served basis through pre-registration only. For conference information contact: Hands On Technology, Pepperdine University, 400 Corporate Pointe, Culver City, CA 90230. Co-sponsored by: Pepperdine University; Computer Using Educators, Inc.; Computer Using Educators, Los Angeles; International Council for Computers in Education and SIG Logo; and the California Regional Educational Television Advisory Council.

LOGO EXCHANGE

Volume 7 Number 4 Journal of the ICCE Special Interest Group for Logo-Using Educators January 1988

Founding Editor
Tom Lough

Editor-In-Chief
Sharon Burrowes Yoder

International Editor
Dennis Harper

Field Editors
Anne McDougall
Richard Noss
Harry Pinxteren
Fatimata Seye Sylla
Jose Armando Valente
Hillel Weintraub

Contributing Editors
Eadie Adamson
Gina Bull
Glen Bull
Doug Clements
Sandy Dawson
Judi Harris
Gary Stager
Leslie Thyberg
Dan Watt

International Council for Computers in Education
Anita Best, Managing Editor
Vincent Elizabeth Barnett, Advertising
Dave Moursund, CEO
Keith Wetzel, SIG Coordinator

SIGLogo Board of Directors
Peter Rawitsch, President
Gary Stager, Vice-President
Ted Norton, Communications
Frank Matthews, Treasurer

Publisher
International Council for Computers in Education

Logo Exchange is the journal of the International Council for Computers in Education Special Interest Group for Logo-using Educators (SIGLogo), published monthly September through May by ICCE, University of Oregon, 1787 Agate Street, Eugene, OR 97403-9905, USA.

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

Contents

ICCE Membership (includes *The Computing Teacher*)

	U.S.	Non-U.S.
	28.50	31.50

SIGLogo Membership (includes *The Logo Exchange*)

	U.S.	Non-U.S.
ICCE Member Price	24.95	29.95
Non-ICCE Member Price	29.95	34.95

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

© All papers and programs are copyrighted by ICCE unless otherwise specified. Permission for republication of programs or papers must first be gained from ICCE c/o Margaret McDonald Rasmussen.

Opinions expressed in this publication are those of the authors and do not necessarily reflect or represent the official policy of ICCE.

From the Editor

What NeXt?

You don't have to be an avid reader of computer magazines to be aware of the introduction by Steve Jobs of a new and exciting microcomputer. Jobs, one of the founders of Apple Computer, Inc., has again formed his own computer company, this time to produce the NeXt machine. If you are interested in the details, you can read reviews in magazines as diverse as *Newsweek* (October 24, 1988) and *Byte* (November 1988.) Suffice it to say that the NeXt machine has an enormous amount of memory, amazing speed, and an unbelievable amount of built in software and reference works.

We won't be seeing NeXt machines in the public schools in the coming academic year. Jobs is initially marketing only to universities. And with a price tag of $6500 it will be beyond most school budgets for some time to come. However, its introduction does point the way towards the next generation of computer systems that students and teachers of the future will have at their disposal.

Whenever there is a new and exciting development in the computer field, I find myself looking both backwards and forwards. A look into the past reminds me how far we have come in only a few years. A look into the future causes me to speculate about where we may be by the turn of the century.

For a moment, look backwards with me. I saw my first microcomputer in the fall of 1978. It was a TRS-80 with 4K of memory and had BASIC built in. I was amazed and astounded. I had worked on large mainframe computers for a number of years and I simply couldn't believe that such a little box was really a computer. I remember clearly deciding right then and there that I simply had to have one of these machines in my classroom — as soon as possible. To make a long story short, I was able to obtain state grant money to get an Apple II Plus computer (with 16K!!!) for the following school year. I spent the summer writing programs in BASIC for my students to use in math class. Although 16K seemed like an enormous amount of space, it didn't take long until I ran out of memory. My plans were simply too grandiose for my machine.

A year or so later, we placed several Apple's in the high school to be used to teach Pascal. Having previously taught Pascal on a large mainframe system, I was constantly frustrated with the lack of memory and slow speed of my now 64K systems. Nonetheless, the ability to program in Pascal on a machine in my own classroom still seemed like a miracle.

About this time Logo became available for the Apple. Of course, I added Logo to my repertoire of programming languages and to the curriculum offered in my school system. As I became more and more involved with Logo, I began to encounter the Logo critics. Not infrequently I heard about how slow Logo was or about its limited memory. Those who claimed to be serious computer users scoffed at those of us who described Logo as a powerful programming language. Had these critics no memory and no vision? Only a couple of years earlier, BASIC seemed quite limited and ran out of memory quickly.

Of course it wasn't long until more sophisticated Logo's arrived on the scene, including very powerful versions like Coral Logo for the Macintosh. Coral Logo extends Logo's ceiling upwards while retaining the low threshold and taking advantage of the friendly Macintosh interface. While benchmark speed tests are not of particular importance to my work with Logo, those who find such matters important need to look beyond the versions of Logo written for that first generation of school computers to get a taste of what the future may hold.

And what of a look forward? No, the NeXt machine doesn't come with Logo — but it certainly would be no problem to tuck a Logo interpreter away among the dictionaries and application software that is part of the package. As more powerful machines like the Apple II GS, Macintosh, and IBM PS/2 series come into the schools, will Logo become more respected among those who focus on issues of speed and memory?

Getting beyond the issue hardware capabilities, what about the future of the Logo culture and learning environment? Will the magnificent machines of the future incorporate Logo-like environments for students to explore, or will they contain only traditional applications software, huge databases, and traditional CAI? Will the computer become just a productivity tool? Will it be used in schools only to teach factual material and raise test scores? Or will new machines incorporate discovery-based Logo-like environments in which students can explore and grow and learn to solve problems of their own invention?

Think how far we have come in the 8 - 10 years since microcomputers first became available and found their way into classrooms. Then, imagine how far we may grow in the next decade. Those of us intimately involved with Logo have a vision of a computer culture in which students control the computer rather than being controlled by it. We want this special vision to be a part of the future — to insure that computers in the schools are used for more than drill and practice or only as tools. As a user of Logo, make it your New Year's resolution to pass on your own special vision to a student or a colleague. Only by sharing our vision can we be sure that it will be nurtured by others and continue to grow.

Sharon Burrowes Yoder
SIGLogo/ICCE
1787 Agate Street, Eugene, Oregon 97403
CIS: 73007,1645 BitNet: YODER@OREGON

Monthly Musings

Interfaces: Transition Zones

by Tom Lough

In December's column, I offered some ideas about transitions and their various characteristics. The focus was on the transition as an action, and how it affected us. This month, I would like to extend those ideas to include a few thoughts about a different aspect of a transition: the vicinity in which it takes place.

Strictly speaking, a transition is a change from one state to another. This could be caused by a variety of factors. One such factor is a change of one's location in space. Most of us have experienced a transition which is dependent on our location. Let me briefly recount one which I remember vividly from my childhood. Growing up on a farm in the beautiful Shenandoah Valley of Virginia, I was riding my bicycle down a dusty country road at twilight. The shadows were lengthening, and the barnyard animals were preparing for an evening of rest. Suddenly, as the road dipped down to a darkening creek bed, I felt it. The temperature of the air changed abruptly from a comfortable one to a chilly one. Without knowing it at the time, I had just experienced what it is like to go through a thermocline, the boundary or surface between two regions of different temperatures. I had sensed the transition from one temperature to another because I travelled from one region to another and passed through the *interface* between them. I turned around and went back up the hill slowly until I experienced the hot fudge sensation of having my upper body in warm air and my lower body in cool air. I had found the interface! I marvelled about that sensation for several days.

The definition of the term *interface* has had an interesting history. I found the term in several dictionaries of earlier years. Prior to the mid-1970's, there was generally just one definition offered, something similar to: a surface (often a plane surface) forming a common boundary between two bodies or spaces. However, starting in the late 1970's, additional definitions began to appear, such as: a common boundary between human beings or systems, and computer equipment or programs which communicate information from one system to another. These additional definitions show clearly the effect of technology and its development. However, all the definitions have a common element, that of a region called a boundary. This region of demarcation serves to distinguish one space or body from another.

Suppose we could greatly magnify the interface between two distinctly different regions. What would we observe? If you believe that you would see a sharp edge, a definite delineation between two regions, you are in for a surprise. A close friend of mine once tried to create an extremely sharp definite edge as part of his Ph.D. research in photogrammetry. He placed a precision-ground razor blade flat onto a piece of photographic paper, covered it with clear glass, and exposed it to light. After developing the photographic paper, he obtained a solid white region separated from a solid black region by what looked like a razor-sharp boundary. However, upon examination under a microscope, the boundary was seen to consist of a narrow band of gray, a cloud of small black and white regions distributed irregularly, yet clearly exhibiting a change in density from the black region to the white one. Within the region of the interface, the transition from white to black was surprisingly neither instantaneous nor definite.

Let's look at a more abstract type of interface. Fractals have been in the news lately. In particular, a set of complex numbers defined by Benoit Mandelbrot displays a curious property of self-similarity. When plotted, the numbers of the so-called Mandelbrot set produce a figure which bears a striking resemblance to recent versions of the Logo turtle! (See Harold Brochmann's excellent article in Sandy Dawson's *MathWorlds* column on page 14 of the May 1988 issue of *Logo Exchange*.) However, as you zoom in on the interface or boundary between the numbers in the set and those not in the set, a surprising thing comes to light. The interface consists of smaller regions with the same shape as the original set. There is no clear boundary as such; instead, what appears to be the edge is composed of fractal areas which, to arbitrary precision, belong to one region or the other. The smaller subregions are blended in a regular self-similar way so that if a point is placed randomly in the edge or interface, it could land in a miniscule area which is associated with either region.

As suggested above, interfaces can be special. If we step away a few thousand meters and take a look at the earth, we note that it can be described in terms of interfaces. Boundaries between water and land, air and land, and air and water are easily distinguished. Within the atmosphere, boundaries between warm, moist air masses and cold, dry air masses are often marked by lines of cloud formation. Are these boundaries also indefinitely defined?

Many scientists recognize that interesting phenomena occur at interfaces. (For example, there is a separate branch of physics / materials science called surface physics which studies microscopic phenomena at the surfaces of materials.) But you do not have to be a scientist to note a variety of interesting phenomena at interfaces. For example, we find the most interesting life forms on earth (including ourselves!) at the interfaces. In fact, more than one person has suggested that such interfaces are necessary for life to be possible. I wonder if an indefinite quality of such interfaces is the important property?

Monthly Musings -- continued

As we turn our thoughts toward our professional lives, the classroom suggests itself as a region of interface, a special place where special things happen for students on their journey through the educational system. And, certainly, Logo is a most interesting and provocative interface between students and computers. These ideas are more in line with the recent definitions for interface provided above. Yet, it is not enough merely to provide these interfaces passively to students. We must be willing to nurture, encourage, and support our students interactively, recognizing that these interfaces are critical mileposts on their journey to becoming. To me, this is an exciting realization!

Finally, let us think about the interface between calendars. At our home each New Year's Eve, our family gathers for a quiet hour of review and projection of goals in various categories. The review consists of examining progress toward goals set last year. The projection includes setting or modifying goals for the coming year. We feel that interface, that boundary time, is special because we make it so. And, having moved through it, marking the transition from 1988 to 1989, we look FD with you to the promise of the New Year.

FD 89!

Tom Lough
Founding Editor
PO Box 394
Simsbury, CT 06070

About the Cover

The drawings on this month's cover were done by students in Judi Menken's classroom in New York City. These six year olds had no more than 6 to 8 months of Logo experience when they created these masterpieces.

Logo Ideas

An Adventure
by Eadie Adamson

This is the first of a series of columns which will focus on the text-processing side of LogoWriter. This month we look at a different approach to word processing through planning and writing interactive stories.

When I read "Reading and Writing Interactive Stories" by Maribeth Henney in the May 1988 issue of The *Computing Teacher*, I realized I had to write a column in response to her wonderful summary of story writing using Scholastic's *Story Tree*. Working with choose-your-own ending stories can be an excellent introduction into word processing. At the same time, you can teach a little about interactive programming.

For anyone who has been much involved with using LogoWriter, some of what Maribeth was describing must have seemed very familiar. One of the LogoWriter intermediate booklets, *Word Adventures*, has a project to get students started writing interactive stories. As I read Maribeth's article, I mentally switched her process to LogoWriter and found myself thinking that this article was a good summary of a delightful idea I've used with LogoWriter. Interactive stories are an excellent curriculum idea for a classroom teacher who knows just a little about LogoWriter and who might be more comfortable working with writing than with graphics. With a few tools to help with the process, developing interactive stories can become a really exciting project.

Here is a description of the process Maribeth described, translated into a LogoWriter project:
(1) The students type one page of a story at a time.
(2) The bottom of the page gives the reader a choice to make.
(3) The student writes the procedures on the flip side of the page which move to the next choices.

There are several possible ways to branch:
- The reader can continue by typing a word or pressing a key.
- The reader can be prompted to type a choice from a list of several pages in order to continue. Each choice will move to another page.
- The writer can ask the reader to signal when ready, then allow the story to move to a randomly chosen page. The random choice can be weighted in favor of a particular branch (see below) if the writer desires.
- The writer can ask the reader for a choice, but then move randomly to a series of pages (allowing another kind of multiple branch).

Preparing for an Adventure-Writing Project

Before beginning the project, it is helpful to connect the idea of branching stories to something most students already know about: the choose-your-own-ending books. Bring several samples to class and have students spend some time working through one of them individually or in groups. If there is oral reading time, choose-your-own-ending books are a delightful way to encourage the active participation of your students.

Next, create a short, simple example in LogoWriter so that the students can get an idea of what they will do at the computer. You can use the samples in the *Word Adventures* book or invent your own. One caveat: Do not make your example elaborate. The purpose is simply to give an idea of how such stories are done on the computer. Although it is more fun to create a more elaborate story, you will probably trigger many stories similar to your example rather than original thinking on the part of your students.

Begin Away from the Computer

This kind of project is ideal for a class with only a single computer or a brief amount of weekly computer time, since much of the planning can be done in class. After introducing the idea, spend time brainstorming beginnings together. Decide whether you want students to work in groups or individually. Set limits for the minimum (and perhaps the maximum) number of branches.

Separate pages can be written away from the computer. The teacher needs to help the students determine how long each page should be: probably only a short paragraph of two to four sentences will fit on each page. Students might begin by writing a longer story, cutting it (literally) into pieces by paragraph, and pasting the pieces on separate sheets. Then they can rearrange the parts and number them accordingly, giving each a special name (later to be the LogoWriter page name.) They also need to decide where and when to offer more than one branch in the story.

Names for pages need to be short, easy words so that the reader has no problem copying them from the screen. Students should use the LogoWriter Reference Manual or Quick Reference Guide to be sure they have not chosen Logo primitives as page names. Although primitives *can* be used as page names, they cannot be used as procedure names. This is where the difficulty arises. If by chance a primitive has been chosen, a synonym must be substituted as the choice for that branch.

Once the branches have been selected, make a story tree. At the top of a sheet of paper, in the center, write the name of the first page. Draw a box around it. Below this box, working left to right, write the names of the pages which are branches from the first page. Put boxes around them. Then draw lines connecting these pages with the first page. Students love to make diagrams and should enjoy this part of the work.

In some circumstances a teacher may prefer to create and duplicate a page with a series of connected boxes which contain the minimum number of branches the students are to write. This page can be given to the students to fill in as they plan their story. With this much advance planning, time at the computer can be spent well.

Incidentally, story development could also be a group project in which different groups of students are assigned different story trees to work on. The whole series of stories can eventually be put on one disk and linked via other pages. Such a project requires more management and planning, but is an interesting way to create a really long adventure. A large, detailed story tree is an absolute necessity in a long project in order to keep track of who was in charge of which pages and how the pages interrelate.

Again, working away from the computer, students can use the *flip* side of their manuscripts to write the procedures for connecting the branches. If there are two choices for the reader they need two procedures on the back of that page, one named with the first word the reader is given as a choice, the other named with the second word. The procedure merely gets the next page. If the entire manuscript is already assembled with a name at the top of each page, the process should be fairly simple.

The GETPAGE Procedure

Most branching of the stories will be done by using a simple procedure which likely resembles this:

```
TO CONTINUE
GETPAGE "next.page <-next.page can
END                    be any name
```

Most children will spontaneously use the same name for the procedure and the page to be selected by GETPAGE. This is not necessary, however. Often students will be happier if a page name tells what happens on that page. Typing LEAVE, for instance, might be better in the context of a particular story, even if its resulting page is named CAUGHT. GETPAGE is merely a command to get the page. There is no magical connection of the page name with the procedure name. It is typing the *procedure* name that activates the procedure and gets the next page. At some point you should make clear to your students the distinction between the page name and procedures on that page.

Computer Time

With the completed first drafts of their pages in hand, students can either work at the classroom computer at their designated times or take the pages with them to the computer lab as a group.

Each student begins entering pages by following some instructions beside the computer. (You might want to use the instructions given at the end of this column.)

Editing or rewriting is best done from printed copies of pages. The printing can take place in the classroom or in the computer lab, whichever fits your schedule best. The hard copy can be marked with changes which are incorporated easily when at the computer.

Logo Ideas -- continued

The Choices

I emphasize with my students the importance of giving the reader clear choices. It is easier to understand what to type if the choices are set off in the text *and* capitalized, like this:

```
-----------------???-----------------
If you want to stay, type
   STAY

If you want to go somewhere else, type
   LEAVE
■
```

Students may have other variations, based on their adventure-reading experiences. The main point is that the choices need to be very clear to the reader.

Some Added Twists to the Computer Story

In addition to using a procedure which responds by getting the next page, there are some other ways of connecting the pages.

Wait for a key to be pressed

If the reader is merely to press a key to get the next page, use a STARTUP procedure on the flip side of the page. A procedure named STARTUP is always run first when a page is loaded. The following STARTUP procedure waits for a key press before moving to another page:

```
TO STARTUP
TYPE [Press any key to continue...]
NAME READCHAR "KEY
CC
GETPAGE "next.page
END
```

TYPE makes text appear in the Command Center, without a carriage return. If you are not clearing the Command Center (CC) as we are here, add TYPE CHAR 13 after the last TYPE command to bring the cursor down to the next line.

NAME READCHAR "KEY simply waits for a key to be pressed. READCHAR reads the character typed on the keyboard. Then Logo moves on to the next task, clearing commands from the Command Center and then getting the next page.

Moving to the next screen after a key press

If the story has more than a single screen of text, a different kind of STARTUP procedure can be used. The following version uses the text primitive NEXTSCREEN to move the text on the page. At the end of this procedure the command TOP moves the cursor back to the top of the text so that the next time the page is used the reader begins at the top. (If you have LogoWriter 2.0, it is better to LOCK a finished page. Then the extra step of moving back to the top of the page can be omitted.)

```
TO STARTUP
TYPE [Press a key to continue...]
NAME READCHAR "KEY
NEXTSCREEN
NAME READCHAR "KEY
CC
TOP
GETPAGE "next.page
END
```

A key press moves to a random choice of pages

Pressing a key might branch to a randomly chosen page from a list of pages. This kind of STARTUP procedure looks like this:

```
TO STARTUP
NAME [LEAVE HIDE SCREAM CLIMB.UP
ESCAPE CAUGHT] "PAGES
TYPE [Press a key to continue...]
NAME READCHAR "KEY
GETPAGE PICK :PAGES
END
```

The PICK procedure looks like this:

```
TO PICK :LIST
OUTPUT ITEM
   (1 + RANDOM COUNT :LIST) :LIST
END
```

PICK chooses randomly from the list named PAGES in the STARTUP procedure. Be sure the pages are specified in STARTUP. Otherwise, PICK may choose from an old list on another page or have no pages to choose at all. This STARTUP procedure is what runs this show and that's where the list of pages belongs. Give your students a model with the list of pages omitted so that they can fill in their own pages.

The choice is not what it seems

PICK can also be used with GETPAGE in a procedure which tricks the reader. The reader can be asked to make a choice, but typing the word gets a random choice of pages, just as the key press did in the previous example. The reader might be asked to choose between STAY or LEAVE. But the two procedures might be the same:

```
TO STAY
NAME [STAY LEAVE CAUGHT ESCAPE HIDE]
   "PAGES
GETPAGE PICK :PAGES
END

TO LEAVE
NAME [STAY LEAVE CAUGHT ESCAPE HIDE]
   "PAGES
GETPAGE PICK :PAGES
END
```

Weighting the choice of pages

It is also possible to weight the choices by including a favorite page name more than once in the list of choices. This is similar to using a weighted set of dice in a game. In the example below the chance is greatest for CAUGHT to be chosen. There is also a better chance that STAY will be chosen:

```
NAME [STAY LEAVE CAUGHT STAY CAUGHT
   ESCAPE CAUGHT CAUGHT CAUGHT HIDE]
   "PAGES
```

To Continue

Next month, I'll continue this exploration of using text in LogoWriter. In the meantime, you can get your students started writing their own adventure stories.

Writing an Adventure Story

```
Choose a NEW PAGE.
Hide the turtle:  HT
Name your page:
   NAMEPAGE "name.of.page
   (Substitute the name of your next page
   for name.of.page.)
Be sure your caps lock key is up (so that
   you can type your story in upper and
   lower case letters).
Press the UP keys.
Begin your story.
When you finish, press the DOWN keys.
FLIP and write your procedure(s) for
   getting the next pages.
FLIP and type NEWPAGE (This will save your
   page and get you a blank page.)
Name the new page with the name of one of
   the next pages in your story.
   (Continue asking for NEWPAGE and naming
   it for the rest of the choices.)
Go back to the first page.
Try your first choice.
Go back again to your first page and try
   your other choice(s).
Check off on your story tree the pages you
   have completed.
Print your completed pages using
   PRINTTEXT80 or PRINTSCREEN (Ask your
   teacher which to use.)

   Finish the next set of pages using the
above steps.  Connect your sets of pages
with a main procedure.
```

Eadie Adamson, Allen Stevenson School
132 East 78th Street, New York, New York 10021

Stager's Stuff

The Stuff We Did Last Summer

by Gary S. Stager

The summer of 1988 marked a remarkable event, the fifth anniversary of the annual Logo Institute. Every summer educators from throughout the world come together for two weeks to learn about Logo and Logo's philosophy of education. Novices and experts are invited and contribute equally to the creation of an extraordinary educational culture.

I have participated in the the past three Logo Institutes; twice as a member of the Institute and three times as a guest speaker at the Logo Institute Conference. The Logo Institute has had such a profound personal effect on me that there I can think of more than a few adjectives to describe the experience: stimulating, fun, intellectually challenging, cooperative, exhausting, therapeutic, creative, sharing, warm, and educational.

Dan and Molly Watt are the founders and directors of the Logo Institute and the adopted parents of the five generations of Institute participants. Their contributions to the Logo community should not be underestimated. Dan and Molly have chosen to use aspects of the writing process as a model for the organization of the Institute. There is the pre-writing stage consisting of leaning about each other and Logo, the writing stage when we develop individual and collaborative projects and the Logo Institute Conference at which we share what we've created with others closely parallels the experience of publication/post-writing. Perhaps the most important goal shared by the writing process and the Logo Institute is to create a meaningful context for learning.

The Samba School

The inspiration for the Logo Institute was Seymour Papert's description of the Brazilian Samba School in *Mindstorms*. The samba school is a real-life example of a non-typical school experience in which all of the participants are actively engaged in learning and contributing to the further development a socially cohesive culture. The samba school is an important part of Brazilian life and does not resemble our schools as much as it resembles large social clubs.

During the year each samba school chooses its theme for the next carnival. The members of the school range in age from children to granparents and in ability from novice to professional artist. Everybody contributes to the creation of the dance, the songwriting, and costume making. All members dance together and as they dance they are learning as well as teaching. Even the experts learn from the less-experienced.

A year's efforts culminate in the twelve-hour carnival. What was learned in the samba schools is not abstract. The

Stager's Stuff -- continued

carnival provides a context and purpose for learning. The carnival is not an adult or children's activity like school, but rather a culture where people are inspired to say to one another, "Let me show you something." The motivation for learning and teaching is intrinsic and what is learned does not follow some prescribed curriculum. Everybody in the samba school is a teacher. The teachable moment is often a spontaneous reaction to the acts of fellow learners.

A Bit of History and Culture

The Logo Institute has been held at Keene State College, Keene, New Hampshire; Union College, Schenectaday, New York; and this past summer at Lesley College in Cambridge, Massachusetts. Lesley was an wonderful site for the Logo Institute. The facilities were excellent and the proximity to Boston, Harvard, and MIT provided numerous opportunities to enjoy a variety of cultural, educational, and culinary experiences. Lesley College was incredibly cooperative and hospitable largely due to Ricky Carter's tireless efforts on behalf of the institute.

The Logo Institute resembles the samba school in the variety of people who attend. There are people who have barely used a computer and Logo veterans such as myself or my fellow columnist, Eadie Adamson. There are Logo Institute rookies and Marianne Schantzenbach who attended the first and fifth Logo Institutes. There are teachers nearing retirement and first-year teachers. Teachers from throughout the United States and around the world attend the institute. I've met educators at the Logo Institute from such exotic locals as, Japan, Australia, Columbia, Brazil, New Guinea, Canada, Switzerland, and Texas. As in the samba school you really have the feeling that the faculty of the Logo Institute learn as much as the students. The distinction between teacher and student are quickly obscured -- a remarkable achievement when you have a faculty that includes Dan Watt, Molly Watt, Ricky Carter, Michael Tempel, Alison Birch, Dorothy Fitch, and Sharon Burrowes Yoder.

The commaradarie of the Logo Institute is quite moving. By the end of the first day most of the participants know each other on a first name basis. Everybody bends over backwards to coach one another and close friendships form quickly. We eat, sleep, and talk Logo for two weeks. This intensity affords us the opportunity to focus on our work and thoughts in ways not possible in the real world.

I find the nicest part of the Logo Institute to be the sense of family that develops. Chuck and Rhonda Rawlings are a fixture at the Logo Institute and their prescence contributes greatly to this sense of family. Chuck and Rhonda are veteran teachers who love people and love teaching. They have attended the past three Logo Institutes. They are energetic, enthusiastic, and incredibly giving. Their sense of humor and ever-present smiles energize their often tired colleagues. Chuck was the institute librarian this year tirelessly assisted the faculty and participants while Rhonda programmed day and night. The Rawlings extended the Logo family by introducing us to their son and daughter-in-law. They joined us on several occasions throughout the institute. I'm sure that I speak for all of the Logo Institute participants, 1986-88, when I hope that Chuck and Rhonda Rawlings will participate in all future Logo Institutes. I eagerly await summertime so that I can attend the Logo Institute and spend more time with Chuck and Rawlings.

A Day at the Logo Institute

The format of the Logo Institute is quite refreshing. Each day begins with a large group session in which topics of interest to the entire institute are explored. Topics have included: Debugging Strategies, Critical Aspects of a Logo Culture and Logo Learning, Understanding Recursion, Logo Around the World, and the development of Logo.

Following the general session, members of the faculty and institute lead a variety of one or two hour sessions catering to a broad range of interests and abilities. Some of these topics are explored for only one day and others form mini-courses which may continue throughout the entire institute. Institute members may attend many of these daily sessions or none of them if they wish to work on individual projects or explore other areas of interest. Examples of mini courses have included: Creating a Logo Toolbox, LEGO TC logo, Critical Aspects of Logo Learning, Beginning Turtle Graphics, Logo and Computer Science, Exploring Language with Logo, Logo Staff Development, Research Issues, Creating Logo Databases, and Logo from Another Planet.

The long Logo Institute day ends with a period for personal reflection, journal writing, and the sharing of programming projects in process. This is an opportunity to help one another, debug programs, suggest project enhancements, and to be inspired by a colleague's work. I have often found myself excited by a project being worked on by a fellow participant and either assisting with the project or suggesting ways to proceed.

After the day's formal activities it is time for dinner. During dinner the conversation often focuses on the day's activities, a programming project, or educational philosophy. My so-called friends even forced me to eat Thai food! Evening activities are left to each person's discretion. We might take in a movie, go shopping, walk around town, listen to a concert, go to a jazz club, visit a museum, venture forth on the now legendary quest for the perfect ice cream, call home, or stay up all-night programming. Past Logo Institutes have included hikes, barbeques, video parties, and swims in country lakes. This year, many of us spent the Sunday between weeks on a four-hour cruise from Boston Harbor to Provincetown, Cape Cod. The sightseeing trip took all day and included a couple of hours in the quaint village of Provincetown. A good time was had by all -- imagine an entire day without a computer!

The Logo Institute Conference

The culmination of the Logo Institute is the Logo Institute Conference. This two-day conference is the Carnival or Publication portion of the Logo Institute. Logo Institute participants who just days ago didn't know what Logo was share their creations and what they've learned with their fellow participants and an extended Logo community.

The Logo Institute Conference is a wonderful celebration of Logo and the joys of education. Logo-using educators and experts from across the the United States are invited to make presentations and share experiences with the members of the Logo Institute. A strong sense of extended family is quickly felt by all in attendance.

Perhaps the greatest part of this event is that it's very much a family reunion. Nearly everybody at this year's conference were alumni of previous years' Institutes. All year long, I look forward to interacting with the friends I've made at other Logo Institutes. I can hardly wait to stay up all night with Mark Lindsay and Al Lewandowski from Michigan debating educational philosophy. Mark and Al are great buddies who drive non-stop to wherever the conference is being held and then spend most of the time arguing with each other.

The party at the Logo Institute Conference has a long established tradition of delightful madness and mayhem. How often does a group of 95 adults (teachers no less) have the opportunity to juggle, blow giant bubbles, sing recursion and turtle songs, play rhythm instruments, perform list processing skits, and share time with people who all love teaching? Imagine watching Dan Watt, "the bubble terrorist", dropping nine-foot soap bubbles from the second floor of the dorms or a few of us bouncing balls off our heads and earning graduate credit at the same time! I still want to know how Ricky Carter's four year old son is going to react when Ricky says he had a tough day at work after he observed his dad participating in this craziness.

The conference isn't just fun and games. This year there were four keynote adresses: Steve Ocko, The History of Technology and LEGO TC logo; Brian Silverman, Experimental Math and Logo; Sharon Burrowes Yoder, Logo and Problem Solving; Dan and Molly ended the conference with a personal reflection on five years of Logo Institute's and twenty years of Logo. There are, of course, a number of other sessions and hands-on workshops available throughout the day. It is quite exciting to see colleagues you haven't seen in a year present what they and their students have accomplished during the past school year.

Heading home from the Logo Institute is always a bittersweet experience. I am saddened to leave my friends, new and old, and am touched by the realization that the schools I am returning to provide a stark contrast to the utopian educational culture created at the Logo Institute. The good news is that Logo Institute participants actually get to experience such an ideal intellectual culture and can influence educational change with the personal knowledge that the vision of *Mindstorms* is possible, practical, and effective.

I hope that you will be able to join the Logo Institute family, next year! Remember, in the words of David Letterman, "You'll leave tired, but it's a good kind of tired."

I'll share some of the Logo projects developed at this year's Logo Institute in a future issue of the Logo Exchange.

Neat Year?

As of yet, it is not clear if there will be a Logo Institute in 1989. The Institute is a huge undertaking for Dan and Molly and the host college. There is some speculation that the Logo Institute's time has past. I sincerely hope the Logo Institute never ends. If you are interested in attending a 1989 Logo Institute, please send a note to Dan and Molly at the address listed below.

Dan and Molly Watt
Educational Alternatives
Gregg Lake Road
Antrim, NH 03440

References:

Papert, Seymour. *Mindstorms: Chilkdren, Computers, and Powerful Ideas*. New York: Basic Books, 1980.

Gary Stager
5 Eastside Avenue, Wanaque, NJ 07465
CIS: 73306,2446, Applelink: K0331

Enjoy a week in Oregon while you learn more about Logo.
Attend the ICCE sponsored workshop:

Logo for Leaders — Beyond Turtle Graphics
Staff: Sharon Burrowes Yoder, Dave Moursund
Dates: July 23rd - July 29th
Location: University of Oregon, Eugene, Oregon
Cost: $450 to cover dormitory room and board, 3 quarter-hours graduate credit, materials, and instruction.

Note: A workshop of similar duration and cost focusing on Leadership Development will be offered the previous week by Dave Moursund.

For more information, contact Dave Moursund or Sharon Yoder at ICCE, 1787 Agate Street, Eugene, Oregon 97403 503-686-4414

Conversations With Logo

Part 1
(as overheard by Michael I. Tempel)

Person: I'm having some trouble with my Logo program.
LogoWriter: What seems to be the problem?
Person: Well, I'm trying to position the turtle at some random place on the screen.
LogoWriter: That's easy enough. Go ahead.
Person: **setpos [random 80 random 80]**
LogoWriter: **setpos doesn't like [random 80 random 80] as input**
Person: Why not?
LogoWriter: Because **setpos** likes a list of two numbers as input. It uses the first number to set the turtle's x coordinate and the second number to set the y coordinate.
Person: I know that. Sometimes it works. For example,

setpos [50 50]

There. The turtle moved up and to the right.
LogoWriter: Of course.
Person: Of course. So when I say **setpos [random 80 random 80]** I'm giving **setpos** a list of two numbers, only I'm asking **random** to pick them for me. Right?
LogoWriter: Wrong. You're giving **setpos** a list of four words, not two numbers.
Person: Huh? What four words?
LogoWriter: **random, 80, random and 80.**
Person: But that's not what I mean. I want each **random 80** to report a number. They're in a list because of the brackets so **setpos** should be happy.
LogoWriter: Well that's not how I do things, but if you want, I can change the way I interpret what you say.
Person: Can you?
LogoWriter: Sure, but there may be some unpleasant side effects that you won't like.
Person: I'll take my chances.
LogoWriter: OK. You asked for it. I'm ready. Try your random position setting now.
Person: **setpos [random 80 random 80]**
LogoWriter: There. Is that what you wanted?
Person: Yes! The turtle moved and you didn't complain. Thanks.
LogoWriter: By the way, what's your program about?
Person: It's a tutorial to explain to people how you work.
LogoWriter: That's a good thing to do. Lot's of folks don't seem too clear about what I'm doing. Can you show me some of it?
Person: Sure. It begins like this:

```
to tutor
print [Forward 50 draws a line.]
print [Try it for yourself.]
end
```

LogoWriter: Ok, let's see it work.
Person: **tutor**
LogoWriter: **I don't know how to draws in tutor**
Person: Wait a minute! What's going on here. This worked earlier today! Now you're complaining that you don't know how to draws! And why did the turtle draw that line on the screen?
LogoWriter: What did you want to happen?
Person: I just wanted that text to appear on the screen. That's why I used the **print** command.
LogoWriter: Oh. Well that's what used to happen, but you asked me to change things so that **setpos** would work the way you want it to.
Person: I didn't ask you to mess with **print!**
LogoWriter: Well, I warned you about side effects, didn't I? I can't do it both ways. We'll have to decide.
Person: Do what both ways? I'm confused.
LogoWriter: I can either evaluate what's inside a bracketed list or not. Before I changed things for you, I didn't evaluate what was inside brackets. That is why **setpos [random 80 random 80]** didn't work. Even though random is the name of a procedure, when it's in a bracketed list I just take it literally as a word. I don't run the procedure. The contents of the list makes **setpos** unhappy.

On the other hand, not evaluating what is inside a bracketed list seems to be what you want when you use **print**. When I evaluated what was inside the brackets (using using the new rules you requested.) I asked the turtle to draw a line because the first two words in the list were **forward 50**, they make a perfectly good command. Then I found a word that wasn't the name of a procedure so I complained. Would you prefer that I just take all those words literally, and not evaluate what I find inside brackets?
Person: Well I suppose so. But wait a minute! Sometimes you do run procedures in bracketed lists. What about when I say **repeat 4 [forward 50 right 90]**? You draw a square.
LogoWriter: I don't run what's inside the brackets, **repeat** does. That's her job. She runs lists. **Print** handles lists differently. He puts their contents on the screen. I don't do anything to bracketed lists – remember, I don't evaluate them. I just make sure that they're delivered to the procedures they're intended for.
Person: Ok. But I still have to set the turtle at random positions for my program to work.
LogoWriter: It can be done. You could...
Person: I know! I could write a procedure, let's call it **setxy**, that takes two numbers as inputs. So I could say **setxy 50 50** instead of **setpos [50 50]**. Then **setxy random 80 random 80** should work. Since the **randoms** aren't in brackets, they'll be run. Each **random** will report a number to **setxy**. Right?

LogoWriter: Right. In fact there are some versions of Logo that have a **setxy** primitive that works just that way.

Person: Ok, here's my procedure:

```
to setxy :x :y
setpos [:x :y]
end
```

Let me try it with some actual numbers first before trying it with **random.**

LogoWriter: Go ahead.

Person: **setxy 50 50**

LogoWriter: **setpos doesn't like [:x :y] as input in setxy**

Person: Gimme a break! What's the problem now?

LogoWriter: It's the same problem. Your procedure handed **setpos** a list of two words that aren't numbers. **Setpos** needs a list of two numbers as input. He's very picky about these things.

Person: I gave him a list of two numbers, the value of x and the value of y. Those were numbers because I used numbers as inputs to **setxy.** Right?

LogoWriter: Wrong. **Setpos** got a list of two words as input. The first word was **:x** and the second word was **:y**. Each of them is a two character word which isn't a number. Since these words were in a bracketed list I left them alone.

Person: I see. So there are two rules here. You don't run procedures if their names are inside a bracketed list and you don't find the values of variables if they're inside a bracketed list. Right?

LogoWriter: Actually there's only one rule, the first one. Did you know that you never really have to use ": "at all? It's just an abbreviation.

Person: I didn't know that. Can you explain?

LogoWriter: Sure. If you say **name 0 "black...**

Person: Wait a minute. I don't know about **name.**

LogoWriter: Then you must know about **make.**

Person: Yes.

LogoWriter: Well it's the same. **name 0 "black** is the same as **make "black 0.**

Person: Why have both?

LogoWriter: Some people prefer one form, some like the other. Some folks like to switch off depending on the context in which the command is being used.

Person: Isn't switching confusing? Don't people get the order of the inputs mixed up.

LogoWriter: Yes.

Person: Well, since I know about **make** can we stick with that?

LogoWriter: If we must. I prefer **name.**

Person: Why?

LogoWriter: Well because . . . Wait, that's another discussion. Let's just use **make** for now. When you say **make "black 0**, you make the word black the name of the number 0. Here are two more examples:

```
make "flavors [vanilla
    [chocolate chip] strawberry]
make "greeting "hi
```

Now **thing** is a procedure that can report an object if you give it's name. For example, if you want to print a greeting on the screen you could type

```
print thing "greeting
```

See, the word **hi** is on the screen.

Person: I see that, but I've never seen the primitive **thing** before. I think I can see what it does though. Let me try this:

```
print thing "flavors
```

Sure, you put **vanilla [chocolate chip] strawberry** on the screen. That's what I thought. **Print** needs an input. **Thing** gets the object with the name we gave it as input and reports that object to **print.**

LogoWriter: Right. Once you give something a name, you can ask **thing** to report what it is by giving **thing** its name as input. You're asking for the thing, or object, whose name is the specified word.

Person: But I always say **print :flavors.** Oh I get it. You said that **:** is just an abbreviation. It's an abbreviation for **thing!**

LogoWriter: Not quite. It's an abbreviation for **thing ".**

Person: Oh sure. We don't say **print : "flavors.** But what about using ":" in a procedure like this:

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

LogoWriter: You could write that procedure as

```
to square :side
repeat 4 [forward thing "side
    right 90]
end
```

and it would work just as it did before.

Person: But what about **:side** after to **square.** Can you substitute

```
to square thing "side?
```

LogoWriter: No. The title line is special. It's not a Logo instruction so using the reporter **thing** in that context isn't right. You use just the word side, but it actually doesn't matter much what punctuation you use. You could write

```
to square "side
    or
to square  side
```

Person: Wait. I thought that using " means the literal word and using no punctuation indicates that you want to run a procedure. Is **side** a procedure in your last example?

LogoWriter: No. But remember that the line beginning with **to** is not an instruction. When I see the word **to** I assume that the next word is the name of the proce-

Conversations with Logo -- continued

dure you want to define. Any words after that on the same line I assume to be the names of inputs to that procedure. I just go by the position of the words on the line. I don't really care about the punctuation.

Person: That seems uncharacteristically sloppy of you. You're usually so precise.

LogoWriter: Yes. I suppose you're right. I guess I should settle on some punctuation for procedure title lines and stick to it. Most people use **:** before procedure input names. Maybe I should just go with that.

Person: Actually, I think I like the idea of using ".

LogoWriter: Why?

Person: Well, when you say **make "green 2**, you're using the word "green" as the name of the number 2. Hmm... I think I see why **name** might be better than **make**. Let me reword that. When you say **name 2 "green**, you're using the word "green" as the name of the number 2. When you write a procedure with inputs you're really doing the same thing.

LogoWriter: Except that the name is only used inside the procedure. It's local to that procedure. **Name** creates global names for use by any procedure.

Person: Yes, I know. When we say **square 50** we're implicitly saying

```
name 50 "side
```

for use in the procedure **square**. If **name** uses a quoted word as input, maybe the same should be true about the names of inputs to procedures.

LogoWriter: Well that makes sense.

Person: That was a long digression. I wanted to know why **setpos [:x :y]** didn't work. You said that it was for the same reason that **setpos [random 80 random 80]** doesn't work. Oh I see! **setpos [:x :y]** is really **setpos [thing "x thing "y]**. The rule is that you don't run procedures that are inside brackets. It's the procedure **thing** that isn't being run.

LogoWriter: Right. When you write **setpos [:x :y]** you just disguise the fact that you're using the procedure **thing**.

Person: OK. Well I think I understand all this, but I still need a way to set the turtle at random positions. **Setpos** wants a list of two numbers so I guess I have to get those numbers from two **random** procedures first and then put them in a list.

LogoWriter: That's right. You could use **list** to do that.

Person: Let's see. . . **list** puts together words or lists into a larger list. How about this

```
setpos list random 80 random 80
```

Great! You moved the turtle and you didn't complain.

LogoWriter: **Setpos** needs a list of two numbers as input. **List** needs two Logo objects of any sort as inputs. The two **random** procedures each give **list** a number so **list** is happy. He puts the two numbers into a list and hands them to **setpos**. Can you fix your **setxy** procedure?

Person: I think so.

```
to setxy :x :y
setpos list :x :y
end
```

LogoWriter: Now try it.

Person: **setpos 60 80**

It works! This has been very helpful. Thanks.

LogoWriter: You're quite welcome. Come back again if you have any other problems.

Person: Goodbye

LogoWriter: **I don't know how to Goodbye**

Person: Oh no!

```
to bye
print [Speak to you soon.]
end

bye
```

LogoWriter: **See you again soon**

When he is not eavesdropping on Logo,
Michael Tempel is Director of Educational Services for
Logo Computer Systems, Inc.
He can be reached at
330 W. 58th Street, Suite 5M
New York, New York 10019
212-765-4780
CIS: 71310,2470

Logo LinX

Twice Upon a Time
by Judi Harris

"Anytwo for elevennis?"

This sentence has suffered Logo inflation. Last year, it might have read,

"Anyone for tennis?"

Next year, if things keep going up, it may read,

"Anythree for twelvenis?"

Now that holiday gift bills are starting to arrive, perhaps inflation is the last thing that you want to remember. Yet it can inspire enjoyable classroom exploits with syllabication, sequencing, and homophones.

Rising to the Occasion

Once Logo inflation hits, "I ate a tenderloin with my fork" becomes "I nined an elevenderloin with my fivek." "Fourscore and seven years ago, our forefathers brought forth" reads, instead: "Fivescore and eight years ago, our fivefathers brought fifth." And so on and so fifth.

Danish comedian Victor Borge first introduced the notion of inflationary words in an effort to match language to economic trends. He reminds us that English "is your language; I'm just trying to use it." Borge suggests that we inflate words as a proactive measure, since inflation (like taxation) is inevitable.

Getting a Rise Out of Them

This presents an interesting Logo challenge. The sound of the first step toward solution is a homophonic one. How many different ways are there to spell each of the number words, 1 through 10? Your students will probably be glad to list the possibilities.

one	two	three	four	five ....
won	to		for	
	too		fore	
	tu			

Now, form a list from these homonyms, output by a procedure called PREINFLATION.

```
TO PREINFLATION
OUTPUT [ONE WON JUAN TWO TO TOO TU
   THREE FOUR FOR FORE FIVE SIX
   SICKS SICS SEVEN EIGHT ATE AIT
   NINE NEIN TEN]
END
```

An accompanying list of the same length can output inflated "values" for each of the words, in order.

```
TO POSTINFLATION
OUTPUT [TWO TWO TWO THREE THREE
   THREE THREE FOUR FIVE FIVE FIVE
   SIX SEVEN SEVEN SEVEN EIGHT NINE
   NINE NINE TEN TEN ELEVEN]
END
```

Inflated Ergo

An INFLATE command can be written to output corresponding inflated list elements.

```
TO INFLATE :WORD.PART
IF MEMBERP :WORD.PART PREINFLATION
   [OUTPUT ITEM (ELEMENT
         :WORD.PART PREINFLATION)
   POSTINFLATION] [OUTPUT:WORD.PART]
 END
```

INFLATE uses an adaptation of Alison Birch's ELEMENT subprocedure, which is the opposite of the primitive ITEM.

```
TO ELEMENT :ITEM :OBJECT
IF :ITEM = FIRST :OBJECT [OUTPUT 1]
OUTPUT 1 + ELEMENT :ITEM
   BUTFIRST :OBJECT
END
```

The superprocedure INFLATED uses these four subprocedures to output *more expensive* words.

```
TO INFLATED :LIST
IF EMPTYP :LIST [OUTPUT "]
OUTPUT WORD ( INFLATE FIRST :LIST )
   INFLATED BUTFIRST :LIST
END
```

Students must supply syllabicated words as input to INFLATE. For example, if a user types

```
PRINT INFLATED [WON DER FUL]
```

the computer will return:

```
TWODERFUL .
```

```
PRINT INFLATED [BE FORE]
```

yields

```
BEFIVE .
```

PREINFLATION and POSTINFLATION resultant lists can, of course, be adjusted to predict inflation at any rate. Who knows? Don Juan may someday be Don Eight. Why not adjust the fable now?

LogoLinX -- continued

Literary Lifts

Inflated words make twoderful stories. Here is an uninflated tale that Mr. Borge supplied. Your students may want to translanine it.

> *Once* upon a time in sunny *California*, there lived a young man named Bob. He was a *second lieutenant* in the U.S. Air *Forces*. Bob had been fond of Anna, his half sister, ever since she saw the light of day for the *first* time. They were both proud of the fact that *one* of their *forefathers* had been among the *creators* of the U.S. *Constitution*. They were dining on the terrace.
>
> "Anna," he said, as he took a bite of a *marinated* herring, "you look *wonderful tonight*. You never looked that lovely *before*. Anna looked *wonderful*, despite the illness from which she had not *recuperated*.
>
> "Yes," repeated Bob, "You look *wonderful* tonight, but you have *two* of the saddest eyes I have ever seen."
>
> The table was tastefully *decorated* with Anna's favorite flowers, *tulips*. They were now talking about Anna's husband, from whom she was *separated*, while on the radio, an Irish *tenor* sang "Tea for *Two*."

Elevated Elegies

Let us finish the tale in inflated terms.

> It was midnight. A clock in the distance struck thirteen. Suddenly, there in the moonlight, stood her husband, Don Two, obviously intoxicninededed.
>
> "Anna!" he blurted, "Fivegive me! I am only young twice, and you are my two and only!"
>
> Bob jumped to his feet. "Get out of here, you three-faced triple-crosser!"
>
> But Anna warned, "Watch out, Bob! He is an officer!"
>
> "Yes, he is two, but I am two, three!"

What inflated stories will your students crenine, given these interdisciplinary tools? Until next time, dear LinXers, three-de-loo!

References

Borge, V. (Performer). (1986). *On stage with audience favorites* [videotape]. New York: Gurtman & Murtha Associates.

Birch, A. (1986). *The Logo project book: Exploring words and lists*. Cambridge, MA: Terrapin, Inc.

Judi Harris, 621F Madison Avenue
Charlottesville, VA 22903

Logo Connections

Leaping to Conclusions with Spreadsheets

by Glen Bull and Gina Bull

This column is about connections between Logo and other kinds of hardware, software, and concepts. Ordinarily we might discuss how data from Logo can be transferred to a spreadsheet, or vice versa. However, this month we would like to discuss how similar concepts can find expression in both Logo and spreadsheets. We have chosen spreadsheets as our basis for comparison because of the (dare we say it?) wide*spread* familiarity with them, but other software such as Hypercard would serve as well.

If a thing is worth doing, it is worth doing poorly.

Learning new things is often only possible through a series of successive approximations. Rarely is a skill perfected the first try. Thousands of tennis buffs enjoy their inexpert weekend games just as much as Bjorn Borg or Jimmy Connors. Millions enjoy chess matches with their friends even though they have not achieved even the lowest national ranking. There are two important ideas here. Often activities worth doing can be enjoyed even if done inexpertly. And, more importantly, most experts begin as novices.

This rule also applies to problem solving activities. If it is not possible to solve a problem completely, solving part of the problem may be a good way to begin. Recently we met a teacher who wanted to calculate the ages of children in her classroom. She had just acquired a spreadsheet and wanted to use it to create a template to do the calculations. The initial format of the spreadsheet that she set up looked like this:

	Date of Birth			Current Age		
	Year	Month	Day	Years	Months	Days
John	1979	1	17	__	__	__
Sally	1978	10	6	__	__	__
Sam	1979	3	17	__	__	__

She wanted to know how to create a formula to perform the computations. Although the problem looks easy, it is a nontrivial task for a novice. To see why, let's look at the calculations involved for John and Sam. John was born on January 17, 1979. Let's assume that today's date is February 20, 1989. The calculation would look like this:

	Year	Mo	Day
Current Date:	1989	2	20
Date of Birth:	1979	1	17
	10	1	3

Through a matter of three separate subtractions we would determine that John is 10 years, 1 month, and 3 days old. Sam's case is a bit more complicated. Sam was born in March:

	Year	Mo	Day
Current Date:	1989	2	20
Date of Birth:	1979	3	17
	10	?	3

When an attempt is made to perform the month calculation (2 - 3), the answer of -1 month results. Clearly it is not possible to be minus one month old; therefore it will be necessary to borrow 12 months from the years column. If this is done, we can determine that Sam is 9 years, 11 months, and 3 days old.

The process of solving this problem in a spreadsheet or Logo would be much the same. First find a simpler problem. In this instance, the problem we suggested was this:

	A	B	C	D
1				
2	Current Date:	1989		
3	Date of Birth:	1979		
4				
5				

In spreadsheet terms, the formula which would be placed in Cell B5 would look like this: (B2-B3). The teacher in this particular instance resisted, indicating that she wanted to know the months and the days as well as years. If you insist on solving the entire problem at once, but don't have the skills for a complete solution, the problem may never be solved. That's where our motto that "if a thing is worth doing, it is worth doing poorly" comes into play.

Let's look at the same problem in Logo. The first thing that we'll do is set the current date. (These procedures were written in LogoWriter, which permits use of upper and lower case. In your version of Logo, it may be necessary to type the following in upper case.)

```
To SetDate
Make "ThisYear 1989
Make "ThisMonth 2
Make "ThisDay 20
End
```

Next we'll create a procedure to calculate the years.

```
To Years :BirthYear
Output :ThisYear - :BirthYear
End
```

Now to use the procedures. We set the date, and print the years:

```
SetDate
Print Years 1979
10
```

This tells us that Sam is 10 years old. (Actually, we know he is only 9 years and 11 months old, but we're approximating, right? The purists in the crowd should remain calm a moment longer. We've done the calculation poorly, but we've done it.) Now let's see if we can refine the process, and add a procedure to calculate months as well as years. Since Sam was born in March, subtracting (2-3) gives us -1 months. As noted earlier, we'll have to borrow some months from the years column. We need a line of Logo code that says "If the birth month is later in the year than the current month, borrow some months from the years column." Here's how we wrote this procedure.

```
To Months :BirthMonth
If :ThisMonth < :BirthMonth
   [BorrowMonth]
Output :ThisMonth - :BirthMonth
End
```

When we wrote in BorrowMonth in the Months procedure, we didn't know exactly how we would perform the operation. We only knew that we needed to borrow some months. When we actually wrote the procedure, we found that it was not complicated:

```
To BorrowMonth
Make "ThisMonth :ThisMonth + 12
Make "ThisYear :ThisYear - 1
End
```

Now we're ready to create a procedure to calculate the age in months and years. To calculate the age, we need to set the date, find the months, and then find the years:

```
To CalculateAge :BirthYear
                      :BirthMonth
SetDate
Make "MonthsOld Months :BirthMonth
Make "YearsOld Years :BirthYear
End
```

Finally, we need a procedure to show the age after the calculations have been made.

```
To ShowAge :BirthYear :BirthMonth
CalculateAge :BirthYear :BirthMonth
Print (Sentence :YearsOld [Years
   and] :MonthsOld [Months])
End
```

The procedure accepts the year and month of birth as inputs, and prints the age:

```
ShowAge 1979 3
9 Years and 11 Months
```

The reason for all this is not to provide a Logo procedure to calculate age, but to make the point that each of the Logo procedures above is relatively simple, even though the problem is complex. For those who were wondering, it is also possible to perform similar calculations in a spreadsheet. The equivalent formula in a spreadsheet we constructed looked like this:

Logo Connections -- continued

```
@IF (C4<C11, C4+12)
```

In this formula, C4 refers to a spreadsheet cell containing the current month, while C11 refers to a cell containing the birth month. The Logo equivalent would be:

```
If :ThisMonth < :BirthMonth
   [Make "ThisMonth :ThisMonth + 12]
```

Because Logo is procedural, it reduces the temptation to try to solve the entire problem in a single (indigestible) chunk of code. However, the same problem solving strategy can be applied to a spreadsheet. The strategy of breaking a problem into smaller chunks is not always as apparent in a spreadsheet, but the same techniques apply. The lessons learned in Logo can be profitably transferred to another domain.

The impulse to attempt to solve a problem in one pass seems to be innate. Even experienced programmers frequently yield to this temptation. Traditional evaluation methods may reinforce this tendency. The motto "if a thing is worth doing, it is worth doing poorly" is not a good strategy on an academic examination. There it is important to solve all the problems completely on the first try. Feedback is an aversive stimulus rather than a problem solving aid on an examination. Red marks typically do not stimulate the student to revisit the problem.

We have not shown you how to calculate the age in days in either Logo or spreadsheet terms. This will be, as they say, left as an exercise for the reader. However, we would point out that if a thing is worth doing, it is worth doing poorly. Therefore, we would suggest that you may want to begin by assuming every month is 30 days, even though that is clearly not the case. Once your Days procedure is working properly, the issue of different days for different months can be added as a further refinement. In Logo this mechanism would probably be implemented as a list:

```
To DaysPerMonth
Output [31 28 31 30 31 30 31 31 30
   31 30 31]
End
```

In a spreadsheet, a look up table would probably be used to solve the problem. Of course, even after this problem is solved, there is always the issue of leap year. Happy computing!

Glen and Gina Bull
Curry School of Education
Ruffner Hall
University of Virginia
Charlottesville, VA 22903
BitNet: Glen GLB2B@VIRGINIA
Gina: RLBOP@VIRGINIA

MathWorlds

edited by Sandy Dawson

Logo in Mathematics Education: What to do with it

by Ihor Charischak, CLIME

About 2 years ago I attended an informal meeting of Logo educators at the National Council of Teachers of Mathematics annual conference, where I volunteered to head a committee that would eventually become an organization called the Council for Logo in Mathematics Education (CLIME). I was motivated by my strong belief that if more math teachers were aware of Logo and its potential, it would contribute to children's having a better understanding of mathematics. I still believe that, but I also realize that knowing Logo is not enough. A knowledge of what to do with Logo in the math classroom is also needed, especially given that most math teachers must, in some sense, follow a specific curriculum. In this month's column, I would like to discuss the issue of what to do with Logo in math education first by sharing a letter that I received from a CLIME member and second by presenting a vision that I would like to see become reality.

The letter is from Mary Waters Schofield who lives in Renton, Washington. She is a computer specialist from St. Anthony's School:

Dear CLIME:

Back in 1983, I saw an hour long P.B.S. show which introduced me to Logo. By that time, we had our own computer at home and two young children. We purchased Commodore Logo (and the necessary disk drive) soon afterward. It was a bargain! Prior to motherhood, I'd been employed as a professional engineer doing mainframe and microcomputer assembly language programming, so I saw Logo first as a new programming language and secondly as a learning tool for children. I felt that not enough was being done in my local school, so I responded to pleas for parent volunteers, and, offered to help in the computer room because of my background. One thing led to another and by the spring of 1985, I found myself leading a group of volunteers in training teachers and other volunteers to use Logo with the K-8th grade students at St. Anthony School. The teacher officially in charge and I agreed the most worthwhile use of the school's 6 computers would be teaching the children to program with Logo. I've been working in the schools computer room at least once each week ever since. Some things have changed in the past three years, but we're still demanding that children program the computers using Logo (LogoWriter now) and I'm still convinced there is no better way the students could use the computers.

My purpose in joining CLIME is to become more aware of publications of Logo lessons/activities for presenting/teaching/utilizing mathematical concepts for grades K-8. I've hopes of setting up a Logo activity for each concept presented in the math text books for grades K-8-something similar to Temple Ary's "Exploring fractions with Logo," which appeared in the June, 1986 issue of *The Computing Teacher*. The LogoWriter materials teach how to use LogoWriter commands, not how to teach specific math concepts (like place value, for example). Most books I've see concentrate on Geometry. I'd like to see a complete series for K-8 with good Logo activities for introducing/exploring/utilizing each concept. Is there anything worthwhile available or about to be published?

As I reflected on this letter, no specific publication came to mind. Then I realized that therc were probably lots of Logo activities or microworlds[1] out there that teachers use to teach important mathematical concepts to their students. What if CLIME could collect these Logo microworlds and make them available to teachers? The idea excited me.

This brings me to my vision: CLIME will become a clearinghouse for Logo microworlds that will help children become better mathematicians. These microworlds will be made available to interested educators,. However, before we can offer these microworlds, we need to have them. *You* are the source of these activities. So, if you have used a microworld effectively with children or know of one, please send the information to me. Include a description of the microworld and your experience with it. If possible, send it to us on a disk indicating which version of Logo it runs on. In return we will be happy to share any of our current microworlds. Just send us a disk mailer and appropriate postage. In the next CLIME column in MathWorlds, we will highlight microworlds that we have available.

Example: A Fraction Microworld[2]

In his article *Turning Mysteries into Problems*, Tom O'Brien (1986) suggests that one way of motivating children to become engaged in learning is by creating a mystery that they eventually turn into a problem that can be solved. Here is a microworld that encourages children to reflect on fractions and problem solve in a meaningful way because their interest has been aroused by a mystery. The following activity would be appropriate for students at about the 5th grade level.

Teacher Note Have your students pretend that they live in Number Town and the currency that is used is fractional. Coins come in the form of 1/2, 3/4, 5/8, etc. (Decimal numbers are unknown in this part of the world!) Have them go to Burgerama to buy a burger (3/4 ¢) and a coke (1/2 ¢). The store uses an adding machine that does not seem to be working properly. (The adding machine appears on the screen.)

The task: Figure out what's wrong with the machine and get it to work properly.

The procedures:

MACHINE: draws the adding machine

ENTER: prompts you to enter the numbers you want to add and prints them in the appropriate windows

ADDER: "adds" the numbers

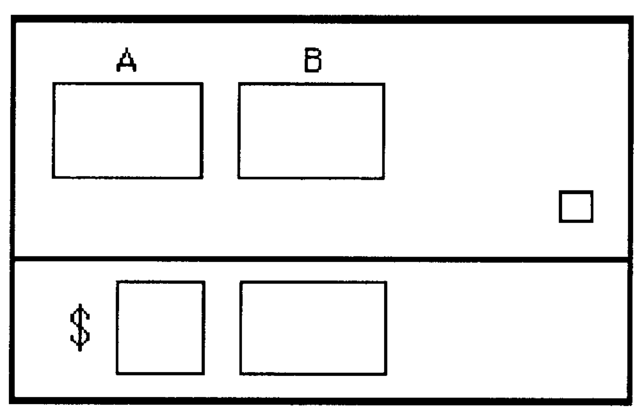
CLONE *location number*: "clones" the fraction in the selected window the number of times specified (creates an equivalent fraction)

CLONE.CHECK: checks to see if the fractions in windows A and B need cloning. If so, the computer will indicate it by turning on the cloning light.

ERFRAC *location*: erases fraction windows

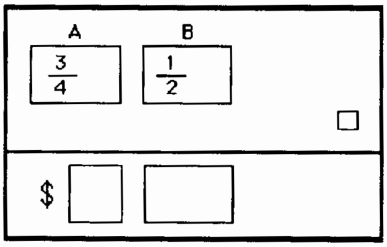
Instructions for using the machine:

1. Type: MACHINE to draw the adding machine on the screen.

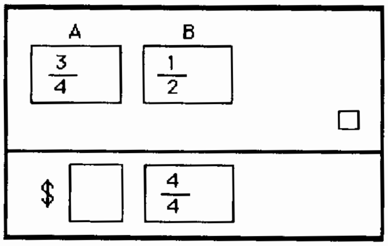


2. Next type: ENTER. The machine will prompt with **Frac #1:** The user enters the first amount 3/4 and presses Return. The computer responds with the second prompt **Frac #2**: and the student types the second amount, say 1/2. The numbers appear in the windows of the machine.

MathWorlds -- continued



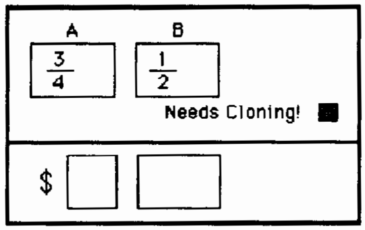
3. Next type: ADDER



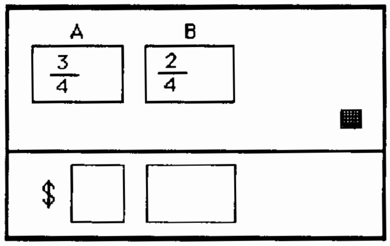
The result will be 4/4 which is not correct. (Children who do not see why it is incorrect can be asked the question: How much money is 3 quarters and half a dollar?) Apparently the adder is not working properly. How did it come up with 4/4? Is it following any particular rule?

Teacher note At this point it is a good idea to have the children figure out how ADDER works. Have the machine try other numbers and see if a pattern emerges. You can start over again by typing ERFRAC which erases all the fractions from the boxes and type ENTER again. (ADDER makes the sum of the two numerators the numerator of the answer, and takes the denominator of the first fraction and makes it the denominator of the answer. For example, ADDER would make 2/3 + 5 /8 = 7/3.) As a general purpose adding fractions procedure, ADDER does not work except for the special case when the bottom numbers are the same. Then ADDER works fine. So before ADDER is applied, the fractions have to have a "clone check" (see if their bottom numbers are the same) to see whether they need to be cloned or not. In Number Town cloning is the process of putting on another fractional costume that has a different numerator and denominator, but maintains the same value. (In more formal language, cloning is the process of finding equivalent fractional forms for a given fraction.)

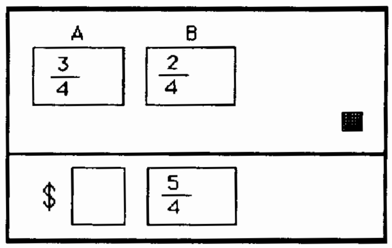
In this example, if you type CLONE.CHECK the machine will blink its light just like a copy machine and print **Needs Cloning!**. That means you have to clone the numbers. (In this case, only 1/2 needs to be cloned.)



Type: CLONE B 2. The fraction in location B (1/2) has its numerator and denominator multiplied by 2. It now turns into 2/4.

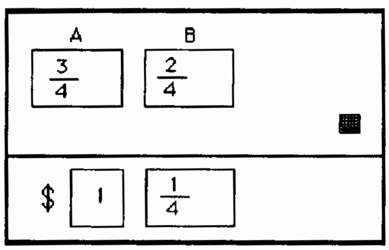


Type: ADDER



The new total (5/4) will now be correct though not in final form: 5/4¢ (five quarters cents) makes sense, but a more common way to describe this amount is: one dollar and a quarter. The procedure MIXED converts quarters (or any other denomination) into whole and fractional parts.

Type: MIXED 1 1/4



In order to "fix" the machine, a program will have to be written that (1) checks for cloning (2) does the appropriate cloning automatically, and (3) converts to a mixed form if needed. An extension of the problem would be to add more entries to the machine so more numbers can be added. Also, the machines capability can be extended to other operations as well.

CLIME

In future columns we will include more examples of math microworlds that we receive from our readers and members. If you would like to participate in helping us build this clearinghouse, please call or write to the CLIME office.

If you are interested in joining CLIME, send the membership fee ($10 in North America, $15, elsewhere) to CLIME, 10 Bogert Ave., White Plains, NY 10606 (914-946-5143). The membership fee entitles you to a subscription to *Clime News.*

[1] A "microworld" is a set of Logo procedures that can help children grasp important ideas. Two classic examples are the Polyspi and Beach microworlds which Bob Lawler describes in his excellent article *Designing Computer Based Microworlds.*

[2] This is activity is part of a larger microworld called *Fraction Arcade* that will be available from Dynamic Classroom Press in the near future.

References

Lawler, Robert (1982) Designing Computer Based Microworlds. *Byte, August issue,* 138-160.

O'Brien, Thomas C. (1986). Turning Mysteries into Problems - Software for Developing Thinking Skills. *The Computing Teacher, March issue,* 39-43.

The Recursive Dragon

by Thomas Bannon

Logo is best known for its graphics routines. This is not surprising since it has some of the most simple yet powerful routines available. Yet its power as a teaching tool and its ability to be considered a serious computer language rests with the natural way in which it handles nested procedure calls and recursion. Unfortunately, too often this aspect of the language is ignored or, even worse, treated as an oddity of the language rather than as one of the most basic, powerful ideas in the language. Logo introduces the student to procedural thinking and programming in a very painless, enjoyable way. Very early on the student is using procedures in such a natural way that he is unaware of what he is doing. At some point, however, the student must become aware of what he is doing. In teaching students to deal with recursion we are making them deal with procedural thinking at a conscious level. This should be a major goal of any serious attempt to teach Logo as a programming language.

A recursive entity is one which is defined in terms of itself. Analogously, a recursive routine is one which calls itself. I have always found writing recursive routines challenging, exciting and fulfilling. A well written one has an undeniable aesthetic charm. They are like polished gems in which we can find unexpected levels of complexity. Unfortunately, many students do not immediately appreciate the power and beauty of recursion. Perhaps the traditional methods of introducing students to recursion are to blame. I have found the typical examples used wanting in one way or another. Calculating factorials has a too obvious iterative solution. Calculating Fibonacci numbers presents recursion at its worst and can only be used as a bad example. In introducing recursion, Niklaus Wirth used a recursive program to draw the Hilbert Curve. I liked the idea of using recursive drawings to introduce recursive programs and remembered that the dragon curve had a recursive definition. I then set about writing a recursive routine to draw the dragon. I wish to share my results with you.

The dragon curve is the answer to a very simple question. What happens if we fold and refold a piece of paper? Every time we fold the paper, we add more creases. If we open the paper and stand it on it's edge, letting the paper form right angles at the creases, the edge of the paper traces a dragon. It is called a dragon because the higher order curves have a strong resemblance to the dragons of oriental art.

The dragon curve is frequently cited as an example of complexity arising from simplicity. Recursive routines

The Recursive Dragon -- continued

also cause complexity to arise from simplicity. Matching the dragon with a recursive routine stresses this theme at different levels.

The dragon diagram was discovered by John E. Heighway and popularized by Martin Gardner in his *Scientific American* column. I was first fascinated by the dragon when I read Dan Rollins article in *Byte* (1983). There are many different ways to make a dragon. The recursive method outlined below is not the fastest but is rather one of the more interesting.

To make a paper dragon, fold a piece of paper once to the left. Open the paper and make the crease a right angle. You have an order-1 dragon (Fig. 1.) From this humble beginning you can soon have a full-fledged dragon. All you have to do is repeat this operation about ten more times. Unfortunately, the number of creases in the paper for an n-th order dragon is 2 to the nth minus 1 which works out to over two thousand for even an eleventh order dragon. Luckily, with a little mathematics and the aid of computer graphics we can soon meet the Dragon.

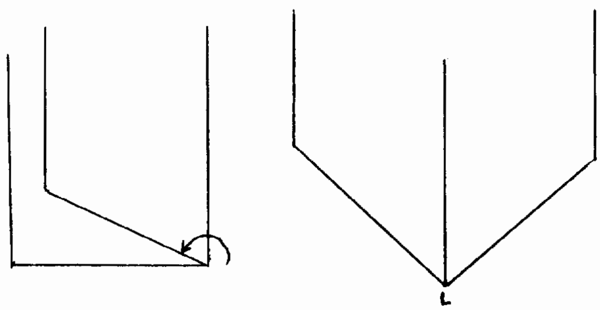


Figure 1

We can describe the dragon by giving the direction of its creases (Figs. 1 to 3.) We can describe an order-1 dragon as L since it has only one crease and its direction is to the left. Folding again to the left, we get two more creases, one more to the left and another to the right. The order-2 dragon is therefore described as LLR. A third fold yields LLRLLRR as the description of the order-3 dragon.

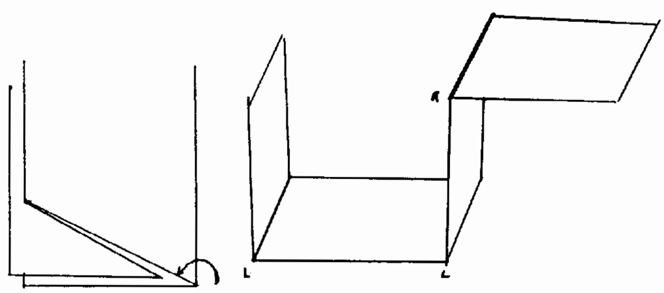


Figure 2

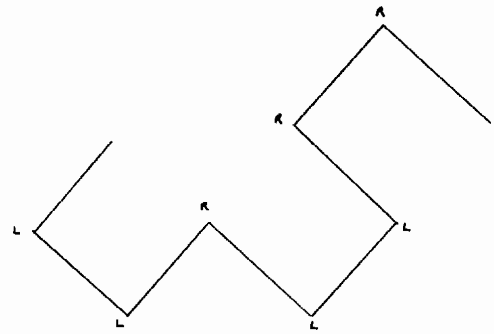


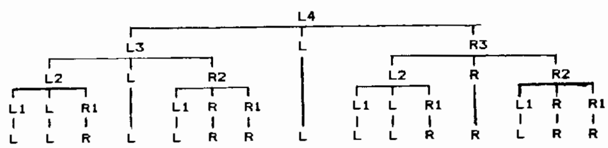
Figure 3

The dragon can be defined recursively in the following manner. Define an L4 dragon as an L3 dragon and an R3 dragon with an L between them. Fine, but what are L3 and R3 dragons? Well, an L3 dragon is an L2 dragon and an R2 dragon with an L between them, and an R3 dragon is an L2 dragon and an R2 dragon with an R between them. Similarly, an L2 dragon is an L1 dragon and an R1 dragon with an L between them. The definitions bottom out at L1 which is L and R1 which is R . An order-n dragon is an L_n dragon. More formally:

$$L_n ::= L_{(n-1)} + L + R_{(n-1)}$$
$$R_n ::= L_{(n-1)} + R + R_{(n-1)}$$
$$L_1 ::= L$$
$$R_1 ::= R$$

Watch it work:

Order 4 Dragon



Why does it work?

Look at the order-1 dragon. Replace its first side with an order-1 dragon with a L crease and its second side with an order-1 dragon with an R crease (Fig. 4.) You get an

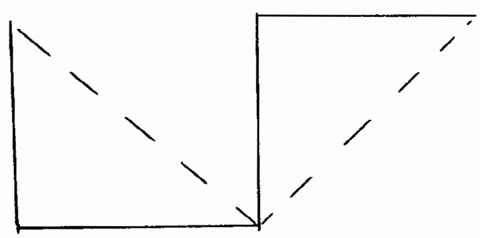


Figure 4

The Recursive Dragon -- continued

order-2 dragon. So the order-2 dragon is made up of two order-1 dragons, one an L-dragon and the other an R-dragon connected by a left turn. Now do the same for each of those two order-1 dragons (Fig. 5). What you get is an order-3 dragon and you are on your way.

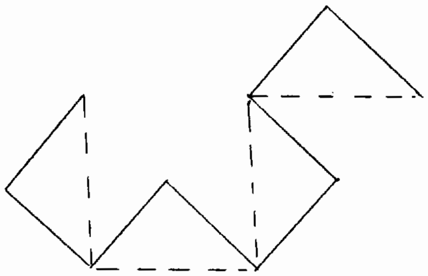


Figure 5

Logo is an ideal language to use for a recursive dragon routine. It supports and encourages recursion. It has extensive, easy to use graphics, and it is available for microcomputers. A language like BASIC would be a poor choice since BASIC does not support recursion, indeed it hardly supports subroutines. I am not saying that the routine could not be written in BASIC, merely that it would be extremely difficult to do so.

In the following listing note that the recursive routine is D. The rest is dedicated to drawing an attractive dragon. The routine is written in Krell Logo for the Apple IIe. It is called by invoking DRAGON followed by an integer giving the order of the dragon you wish to draw. For example, DRAGON 8 would cause an eighth order dragon to be drawn.

```
TO DRAGON :ORDER
INIT (-90) 50
HT
SETH 90 + 45 * :ORDER
SCALE :ORDER 150
D :SZ :ORDER "L
END

TO D :S :O :LR
IF :O = 0 FORWARD :S STOP   (line 1)
D :S (:O - 1) "L            (line 2)
IF :LR = "L THEN LEFT 90    (line 3)
IF :LR = "R THEN RIGHT 90   (line 4)
D :S (:O - 1) "R            (line 5)
END

TO SCALE :ORDER :SIZE
IF :ORDER = 0 THEN MAKE "SZ :SIZE
STOP
SCALE (:ORDER - 1)
   (:SIZE / 2 * SQRT ( 2 ) )
END

TO INIT :X :Y
DRAW
PENUP
SETXY :X :Y
PENDOWN
END
```

Let's consider how routine D would respond to a call to draw a level-2 L dragon. The first task that D would perform would be to suspend operations while it called itself to draw a level-1 L dragon (line 2). To draw a level-1 L dragon we must draw a line segment, make a left turn, and draw a second line segment. We can accomplish this task neatly by having each of the calls for a level-0 dragon draw a line segment (line 1). So the call for a level-1 L dragon will cause a line segment to be drawn (line 2), a left turn to be made (line 3), and a second line segment to be drawn (line 5). On the return, the level-2 L dragon call will continue at line 3. Line 3 will cause a left turn. Line 5 will then cause the present routine to call itself to draw a level-1 R dragon. The level-1 R dragon will then be drawn in a similar fashion to the level-1 L dragon above (line 4 will cause a right turn). The level-1 call will then end and cause a return to level-2. The level-2 call will then end.

To review, line 2 causes a segment to be drawn, a left turn taken, and a second segment to be drawn. Line 3 causes a left turn. Line 5 causes a line to be drawn, a right turn taken, and another line to be drawn. The result is a level-2 L dragon. A level-3 L dragon will be drawn by first drawing a level-2 L dragon, making a left turn, and then drawing a level-2 R dragon. To draw level-3 dragons we must merely draw level-2 dragons and connect them. To draw level-2 dragons we merely draw level-1 dragons and connect them. Since we can draw level-1 dragons, we can draw level-2 dragons. Since we can draw level-2 dragons, we can draw level-3 dragons. And so forth. This is the essence of the idea of recursion.

The purpose of SCALE and the SETH in DRAGON is to scale and orient the curves so that all orders stay on the screen. We saw that we can change an order-1 dragon to an order-2 dragon by replacing its sides with order-1 dragons. In effect the sides of the order-1 dragon become the hypotenuses of the isosceles right triangles whose legs form the order-2 dragon (Fig. 6). A little geometry will show that by making the sides of the order-n dragon equal to SQRT(2)/2 times the sides of an order-(n-1) dragon, the legs of the isosceles right triangles will be correctly scaled and the new dragon will be approximately as large as the old one. Also by increasing the initial drawing angle by 45 degrees in the clockwise direction each time we increase the order by one the dragon will be oriented the same way

The Recursive Dragon -- continued

each time and not spiral off the screen. With a little experimentation an appropriate starting length and orientation can be achieved. I start the order-1 dragon in the upper left-hand part of the screen with an initial starting direction of S 45 E and an initial length so that it fills about 2/3 of the screen. This generally gives good results but some people have complained that it makes the dragon look like a poodle.

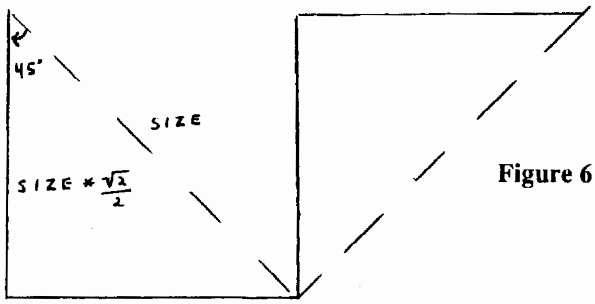


Figure 6

Once your students have conquered the dragon they should be ready for anything, well almost anything. They will have in their conscious possession a very strong tool, recursion.

Bibliography:

Davis, Chandler and Donald E. Knuth "Number Representations and Dragon Curves-I," *Journal of Recreational Mathematics*, April 1970, page 66.

Gardner, Martin "Mathematical Games," *Scientific American*, April 1967, page 118.

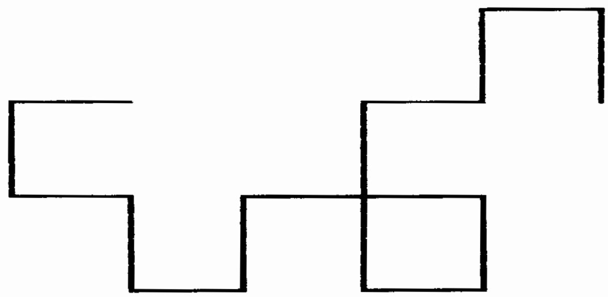
Land, Bruce R. "Dragon," *Byte*, April 1986, page 137

Rollins, Dan "A Tiger Meets a Dragon," *Byte*, December 1983, page 457.

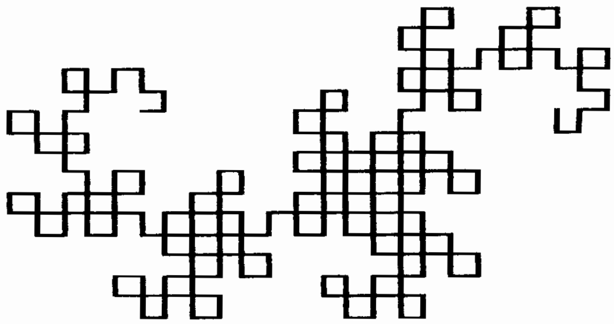
Thornburg, David D. *Beyond Turtle Graphics*, Addison-Wesley, pages 29-56.

Wirth, Niklaus *Algorithms + Data Structures = Programs*, Prentice-Hall, Inc., pages 125-133.

Thomas Bannon
87-57 252 Street
Bellerose, New York 11426



Order 4 Dragon



Order 8 Dragon

Order 14 Dragon

Little Kids and Logo

Turtle Tool Kits: The Tricks are in the Bag
by Leslie Thyberg

One of the concerns I hear most frequently when I am working with school teachers is whether or not little kids can handle the complexities of Logo. "Of course they can!" is my unwavering response. But, there are strategies you can utilize as a teacher to make your students' acquisition of Logo skills go more smoothly. The subject of this month's column is the use of some simple on-computer utilities. Type the ones that you like and save them in a TOOL KIT file.

Turtle Tricks on the computer

Drawing and moving the turtle: Easy commands:

Depending on the version of Logo you are using, I have found the use of the letter "D" for draw to be a simple metaphor for students to understand that when they are in *draw mode* the turtle immediately carries out their commands. By placing a green (g = GO) press-apply dot on the D key, the amount of time devoted to hunting and pecking the letters for DRAW or CLEARSCREEN is significantly reduced. It also gives the students meaningful terminology to speak about their Logo experiences. I labelled this particular set of tools as my HI file. (Hi, for greeting or starting off Logo conversations with the turtle).

```
                Brings the turtle home and
TO D    <-      clears screen
DRAW    <-      (or CS) depending of version
END             of Logo being used
```

In the same set of utilities, I added some turtle movement tools. Sometimes kids have a hard time grasping just how small a turtle step is (equivalent to about one millimeter). How many times have you asked your students to do an introductory exploratory activity such as discovering how many turtle steps it is to the top of the monitor screen? Inevitably while coaching a child to type in a "really big number" they biggest one they can initially conceive of if 9. The following movement tools will help considerably, regardless of whether the obstacle is estimation or a simple typing skill. Instead of typing out the whole word, FORWARD, or its abbreviation, FD, I would have the children use just F or B for forward and backward movement. The F and B procedures are movement tools to create large turtle steps.

```
TO F :INPUT
FORWARD :INPUT * 10
END

TO B :INPUT
BACK :INPUT * 10
END
```

Compass Directions

Another part of my tool kit is the use of compass commands which point the turtle to the desired heading. I have found these simplified commands for compass direction to be equally useful for social studies, weather and science lessons, and an easy way of interweaving Logo into other content areas.

```
TO N            TO S
SETH 0          SETH 180
END             END

TO E            TO W
SETH 90         SETH 270
END             END

TO NE           TO SOE
SETH 45         SETH 135
END             END

TO SW           TO NW
SETH 225        SETH 315
END             END
```

Welcome to Logoland

This is a program a nine year old student wrote for me to use to greet my younger students each time they came to work at the computer. Embedded in it is D for draw - a quick way of clearing the screen which also helps the user differentiate between DRAW mode and EDIT or TEACH modes (especially helpful for those of you who do not have Logowriter).

```
TO HI
NODRAW
PRINT [WELCOME TO LOGOLAND]
PRINT []
PRINT [I AM GLAD TO SEE YOU TODAY]
PRINT []
PRINT1 SENTENCE [WHAT'S YOUR NAME?] "
MAKE "NAME REQUEST
PRINT (SENTENCE [OKAY -] :NAME
      [HERE WE GO!])
PRINT []
PRINT [TYPE D FOR DRAW]
END
```

Slow down the Turtle!

Sometimes what happens for little kids with Logo is the turtle moves so very quickly that they don't always detect its movement or directionality. I was pleased as a classroom teacher to find a solution in Dan and Molly Watt's article, entitled "Starting Logo with Young Children? Slow Down the Turtle! *National Logo Exchange*, September 1986, pp. 3-5.

To slow down the turtle, type:

Little Kids and Logo -- continued

```
TO F :STEPS
REPEAT :STEPS [FORWARD 10]
END

TO B :STEPS
REPEAT :STEPS [BACK 10]
END

TO R :TURNS
REPEAT :TURNS [RIGHT 10]
END

TO L :TURNS
REPEAT :TURNS [LEFT 10]
END
```

While this allows the students to retain access to standard Logo commands at all times, its downfall is that it limits all moves and turns to 10-step and 10-degree increments. A more elaborate variation of the preceding examples is to use the following (also developed by Dan and Molly Watt):

```
TO SLOWFD :DISTANCE
REPEAT (QUOTIENT :DISTANCE 10)
   [FORWARD 10]
FORWARD (REMAINDER :DISTANCE 10)
END

TO SLOWRT :ANGLE
REPEAT (QUOTIENT :ANGLE 10)
   [RIGHT 10]
RIGHT (REMAINDER :ANGLE 10)
END
```

Microworlds

Microworlds are in essence 'task domains' or 'problem spaces' designed for virtual, streamlined experience. These worlds encompass objects and processes that we can get to know and understand. The appropriation of the knowledge embodied in those experiences is made possible because the microworld does not focus on 'problems' to be done but on 'neat phenomena' - phenomena that are inherently interesting to observe and interact with. (Lawler, 1982)

The POLYSPI microworld is a classic example of a playground for exploring the powerful idea of stepping variables:

```
TO POLYSPI :DISTANCE :ANGLE :CHANGE
FORWARD :DISTANCE
RIGHT :ANGLE
MAKE "DISTANCE :DISTANCE + :CHANGE
POLYSPI :DISTANCE :ANGLE :CHANGE
END
```

(Try inputs such as POLYSPI 10 60 3 or POLYSPI 12 65 5) In a future article I will address the wonderful potential of microworlds.)

Controlling the turtle

Children need to develop a degree of proficiency at being able to effectively manipulate the turtle on the screen. Using clear acetate Turtle Mazes or Race Courses, playing a Turtle Target game (such as Dan Watt's Shoot and Pick a Point games in Watt, p. 25, 205ff), having students trace around Colorforms which stick readily to the screen, or even using acetate dot-to-dots (from primary workbooks or coloring books) are all ways of exploring and controlling the turtle's environment.

Instant Logo

Any effective Instant Logo program should have the following characteristics: capability for entry of any number; appropriate turtle step length; appropriate turtle step speed; visible results with small numbers; messages which tell what is happening on the screen; meaningful error messages; easy exit and resistance to crashing (Cochran, 1986).

Dan Watt has an interactive project called QUICKDRAW (Watt, p. 225ff) which is an example of a turtle drawing activity for young children. There are numerous other techniques for making Logo more workable for children such as reading a single key press (Cochran and Glen Bull, *NLX, 1986)* and MECC EZ Logo. Time does not permit a detailed discussion of each area. Instead, the preceding list of examples is intended to provide a small sample of the possibilities available to the Logo teacher. Next month's column will look at some specific curricular activities using Logo microworlds in the content areas in the classroom with some of the tool kit tactics listed in this article.

References

Cochran, Paula and Bull, Glen. (1986). *The Issue of Instant Logo.* Paper presented at the Logo '86 meeting of the MIT Epistemology Group. Boston, MA.

Cochran, Paula and Bull,Glen. (1986, Dec.) Special Talk: Touch Tools. *National Logo Exchange*, pp. 18-20.

Lawler, R.W. (1982, August). Designing computer-based microworlds: Well-designed Logo procedures can help children grasp ideas of intrinsic interest, *Byte*, pp. 138-160.

Thyberg, L. (1988, Mar.) *Little Kids and Logo.* Paper co-presented with Michael Tempel at the annual N.A.M.E. Conference, Somerset, N.J.

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

Watt, Daniel and Watt, Molly's. (1986, Sept.). Starting Logo with Young Children? Slow Down the Turtle! *National Logo Exchange*, pp. 3-5.

Dr. Leslie F. Thyberg, Chatham College
Woodland Road, Pittsburgh, PA 15206

Assessing Logo Learning in Classrooms

IV. Mathematics of Turtle Geometry: Using Mathematical Thinking

by Dan Watt

This is the fifth of nine columns based on a research project which Molly Watt and I have been carrying out with support from the National Science Foundation, "Exploratory Research on Critical Aspects of Logo Learning." In this project, we collaborated with a group of experienced Logo teachers to identify critical aspects of Logo learning and group them under eight headings which were listed in September's column.

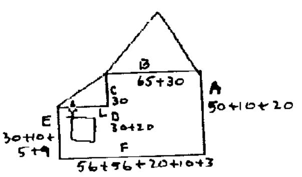
This month I will discuss each of the subheadings included in the fourth cluster of critical aspects, Mathematics of Turtle Geometry -- Using Mathematical Thinking, illustrated with examples of student work. Last month's column dealt with Mathematics of Turtle Geometry -- Using 360 Degrees. For a fuller sense of what we mean by critical aspects of Logo learning, and our rationale for this approach to assessing Logo learning, please read the September '88 column in this series.

What do I mean by "using mathematical thinking"?

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

For an example, let's see how Kathy, a fourth grader, drew the frame of a house.

Example 1: Kathy's House (Grade 4):




```
TO HOUSE.6
PU
BK 30
BK 20
RT 90
FD 30
FD 20
LT 90
PD
FD 50 }
FD 10 } A
FD 20 }
LT 90
FD 65 } B
FD 30 }
LT 90
FD 30 C
RT 90
FD 30 } D
FD 20 }
LT 90
FD 30 }
FD 10 } E
FD 5  }
FD 4  }
LT 90
FD 56 }
FD 56 } F
FD 20 }
FD 10 }
FD 3  }
...
...
AND SO ON ...
END
```

Figure 1

Kathy's approach to Logo drawing is to try out a series of steps in direct mode, and write down each step in her notebook as she goes along. When her picture looks the way she wants it to, she copies all the steps listed in her notebook, and adds them to her procedure, HOUSE.6, using the Logo editor. In this project Kathy used visual problem solving (Kull, 1986) to decide how far to move the turtle. She used a combination of large and small steps until she got the turtle where she wanted it. Then she turned the turtle 90 degrees to go in another direction. If we add up the forward steps in Kathy's procedure — which Kathy almost certainly did not do — we can see that her figure is slightly off: the total heights of the two sides do not quite match, although the lengths of the top and bottom do (See handwritten notes, Figure 1).

Although visual problem solving is perfectly appropriate for many Logo tasks, Kathy did not use mathematical knowledge that might have been helpful to her, such as how to add, or use mathematical reasoning to determine that the two sides of a house should be exactly the same height, or to calculate how far the turtle had turned. Kathy's window is askew because her visual problem solving failed (due to an inconsistency in Logo graphics on the Apple computer). As you can see from the figure, the turtle *looks* as though it is heading straight up, although its actual heading is slightly to the right of vertical.

After examining printouts of this project at several stages, Kathy's teacher asked her to change her way of working: he suggested that after she got the turtle where she wanted it, she add up the forward steps, and use the total number of steps in her procedure. He also showed her how to use the HEADING command to determine the turtle's true heading. From that point on, Kathy's work became easier for her to carry out and easier for her teacher to understand when he looked at it!

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

However, to say that students *can* learn to use certain intellectual tools while using Logo does not mean that all students *will* learn to use them, or that they will apply ideas used in one setting to other settings in which they might be useful. There are many mathematical ideas embedded in Logo. A first step for a teacher, in helping students acquire useful mathematical thinking strategies is to be aware of

Assessing Logo Learning -- continued

what those strategies might be, and when they might be useful for particular projects. The second step is to observe whether students are using them. Finally, a teacher can utilize specific teaching and intervention strategies to help students learn and use particular ideas. Chapter 3 of *Mindstorms* (Papert, 1980) illustrates a number of the most useful and important mathematical ideas built into Logo by its developers. Others are identified in the Final Report of the Brookline Logo Project (Papert, et. al, 1979). In the rest of this column, I will identify just a few specific ways that the students involved in our collaborative research project used mathematical thinking in their Logo work.

Estimating angles and distances

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

One subskill of estimation is understanding units: how big is 1 turtle step, 10 turtle steps, 100 turtle steps. How far does the turtle rotate when you turn right 10, right 90, and so forth. An understanding of 360 degrees and its factors (discussed in this column last month) is extremely helpful in estimating rotations. Another subskill is having a sense of order of magnitude: to move the turtle forward from point A to point B will you need an input of 10, 100 or 500 turtle steps? Vinh's and Martha's Computer project, (Example 2) shows a good ability to estimate turtle steps visually, as well as the use of mathematical reasoning to make use of their estimates in solving problems.

Combining steps

This is one of the simplest mathematical strategies used by Logo students, although we have already seen that it was not used by Kathy (Example 1 above). A sequence of forward steps, Forward 10 Forward 20, Forward 30, can be replaced by their sum, Forward 60. You can similarly combine a sequence of Back, Right or Left commands. The difficulty comes when you have Forward and Back, or Right and Left in the same sequence. Then you have to use the idea of inverses, to subtract all the Back steps from the Forward steps, or all the Right turns from the Left turns to get the correct total. Douglas' PacMan drawing (Example 3) shows a rather complex example of how useful this thinking can be.

Using inverses: right/left, forward/back

Students who recognize that forward/back and right/left can be used to undo each other, or backtrack over a completed part of a drawing, have learned a valuable problem solving tool for turtle geometry, as well as an important mathematical abstraction — the concept of an inverse, a procedure or operation that reverses or undoes another (such as + and -, right and left, forward and back, penup and pendown). Douglas' PacMan (Example 3), Jerry's Tree (Example 4), and Sheila's Face (Example 5) show some possible uses of right/left and forward/back as inverses.

Using specific knowledge about particular shapes

A student who understands the special properties of a square, rectangle, triangle or circle, can use these to solve problems much more quickly than someone how has to figure out each step visually. Vinh's and Martha's Computer (Example 2) shows a good use of the properties of squares and rectangles. Brad's Doughnut (Example 6) uses special knowledge about the radius of a circle to center one circle inside another.

Using right/left symmetry

Right/left symmetry is a special kind of inverse relationship which is extremely useful in solving many Logo problems and simplifying others. To make one figure symmetrical to an existing figure, simply reproduce all the commands in the first figure, replacing each rotation by its inverse. Or, within a symmetrical figure such as a tree or person, make one side symmetrical to the other, by using the same commands, with inverse rotations. Jerry's Tree (Example 4) is a nice example of this.

Using similarity

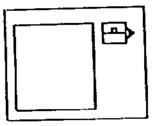
To make the same shape with different sizes, you hold all the angles constant but change the sizes proportionally. This can be quite simple to do using regular figures such as squares or equilateral triangles. Vinh's nested triangles (Example 7) illustrate this nicely. To create similar shapes with more complex figures, proportional variables are required. (I refer you back to Diane's Sailboat, in November's column on Using Variables, for an example of this.)

Using transformational geometry

This involves figuring out how to redraw a given shape in a new position, rotated into a new orientation, changed in size, or flipped symmetrically about a given line. These techniques are useful in mastering turtle graphics, and they are also an important part of many elementary and middle school math curriculums (this is more often true in Canada than in the US). Vinh used a slide/rotation/slide/rotation sequence (probably without thinking of it that way) to position his second set of nested triangles (Example 7).

Examples

Example 2: Vinh's and Martha's Computer (grade 5).

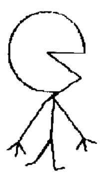


```
TO COM
 M1
 C1
 C2
 M2
 C3
END

TO M1
 PU BK 30 LT 90 FD 80 RT 90 PD
END

TO C1
 REPEAT 4 [FD 85 RT 90]
END

TO C2
 PU LT 90 FD 10
 RT 90 PD FD 100
 RT 90 FD 150
 RT 90 FD 110
 RT 90 FD 150
 RT 90 FD 10
END

TO M2
 PU FD 85 RT 90
 FD 105 PD
END

TO C3
 FD 25 RT 90 FD 20
 RT 90 FD 25 RT 90
 FD 20 BK 10 RT 90
 FD 10 LT 90
 REPEAT 4 [FD 5 RT 90]
 RT 90 FD 15
END
```

Figure 2

Figure 2 shows an early stage of a project that eventually became much more elaborate. I chose it to illustrate the way these students estimated sizes, and used their knowledge of the properties of rectangles. Procedure C1 draws the inner square (the monitor screen) with a size of 85. C2 draws the outline of the computer itself, using a rectangle of 110 by 150. Notice how the last command of the C2 procedure, FD 10, completes the first side of the rectangle by adding 10 turtle steps to the FD 100 used to start drawing the rectangle.

Example 3: Douglas' PacMan (grade 4).



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

Figure 3

Douglas' design shows a striking use of both right/left inverses and combining steps, to draw the head of his PacMan. I've included just part of his procedure. Look at the numbers he used for his turtle rotations: RT 56, followed by LT 112, followed by RT 146. At first they might seem unrelated choices, determined by estimation and experimentation. But if we notice that LT 112 is the same as LT 56 + LT 56, and that RT 146 is RT 56 + RT 90, we can infer that Douglas used right/left inverses and combining steps to make his design come out exactly the way he wanted it to look. I don't know whether he calculated these numbers in his head or on paper, or whether he used the uncombined numbers in his original design, and combined steps later. Either way, it's a powerful example of mathematical thinking embedded in a Logo design, and of the kind of detective work a teacher might need to do to uncover it!

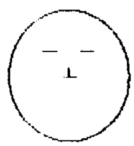
Example 4: Jerry's Tree (grade 4).

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

Figure 4

Jerry's Tree (part of a larger outdoor scene) offers a straightforward, and fairly commonplace use of forward/back and right/left inverses to draw a symmetrical tree.

Example 5: Sheila's Face (grade 4).

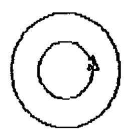


```
TO FACE
 CIRCLE NOSE EYES
END

TO CIRCLE
 REPEAT 360 [FD 1 RT 1]
END

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

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

Figure 5

This is another example using forward/back inverses and combining steps to create a symmetrical face. The radius of Sheila's Logo circle is almost exactly 57 turtle steps. Sheila's Nose procedure moved the turtle to the center using steps of FD 50 FD 14 BK 7. I don't know if Sheila used the turtle to measure the diameter of the circle experimentally before making the Nose procedure, or simply used a combination of estimation and experimentation until the picture looked right to her. Either way, it's a good combination of mathematical strategies.

Example 6: Brad's Doughnut (grade 4).



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

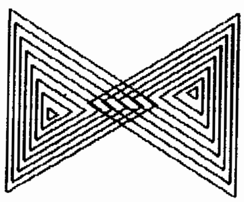
Figure 6

Brad's Doughnut uses a tool procedure, Larc to draw circles. Larc's first input is the radius, it's second input, the number of degrees. Brad's procedure draws centered circles by starting with a circle of radius 40, moving 20 turtle steps towards the center, and drawing a second circle with radius 20. If Brad does not yet fully understand the

Assessing Logo Learning -- continued

properties of circles, this little project offers an opportunity for teaching him about circles and radii.

Example 7: Vinh's Nested Double Triangles (grade 5).



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

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

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

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

Figure 7

Vinh's triangles demonstrate a very simple use of similarity: all equilateral triangles are proportional. The side lengths used in procedures TRI - TRI8, vary from 150 to 10, with decreases of 20 between successive triangles. But Vinh's work is more interesting for its use of slides and rotations to position figures. Each of the internal move procedures, M2 - M8, uses exactly the same sequence (including right/left inverses) to move the turtle from one triangle to the next. The procedures M1 and M1A that move the turtle before drawing each set of triangles demonstrate the use of transformational geometry ideas (slide/turn/slide/turn and so on), along with right/left symmetry. M1A moves the turtle home, then reverses the angles used in M1, to move the turtle to the right side of the screen. Then, because the VU procedure has all triangles turning to the right, Vinh had to move the turtle 150 steps forward and reverse its direction, to begin the design with the turtle in the upper right hand corner.

Although Vinh's mathematical thinking seems rather sophisticated, both in this example and in Example 2, he clearly has a lot to learn about using procedures effectively in a modular fashion, (see October's column) and is probably ready to learn to use variables as well (see November's column).

A Closing Thought

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

References

Kull, J. (1986). "Learning and Logo," in Campbell, P. and Fein G. (eds.), *Young Children and Microcomputers*, Prentice Hall, New York NY.

Papert, S. (1980). *Mindstorms: Children Computers and Powerful Ideas*. Basic Books, New York NY.

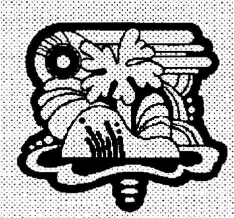
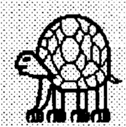
Papert, S., diSessa, A., Watt D., and Weir, S. (1979). "Final Report of the Brookline Logo Project, Part II". *MIT Logo Memo # 53*. MIT Logo Group, Cambridge, MA.

Pea, R., Kurland, M. and Hawkins, J. (1987) "Logo and the Development of Thinking Skills," in Pea, R. and Sheingold, K. (Eds.) *Mirrors of Minds: Patterns of Experience in Educational Computing*. Ablex Publishing, Norwood NJ.

The work described here was conducted at Education Development Center (EDC), 55 Chapel Street, Newton Massachusetts, and supported in part by the National Science Foundation under grant # MDR 8651600, Exploratory Research on Critical Aspects of Logo Learning. The ideas and opinions expressed are those of the author and do not necessarily reflect the views of EDC or the National Science Foundation.

Dan Watt
Educational Alternatives
Gregg Lake Road
Antrim, New Hampshire 03440

• Glen Bull • Tom Lough • Dennis Harper
in conjunction with the
University of the Virgin Islands
Present

Logo in Paradise
U.S. Virgin Islands

A two-week Logo workshop/course will be offered in the Virgin Islands from July 30 to August 12, 1989. The course content will include instruction in topics such as Logo Writer, Lego Logo, Use of Logo with Apple Works, Parallels between Logo and HyperCard. Classroom applications discussed will include subjects such as: use of Logo adventure stories in writing class, use of Logo in science and robotics, use of Logo to generate poetry, parallels between Logo functions and mathematical expressions, and use of Logo with the physically handicapped.

The location of the course offers participants and their families an opportunity to see a unique part of the United States. The course will take place on three different U.S. Virgin Islands located in the eastern Caribbean. The workshop will begin with six nights on St. Thomas. The next three nights will be spent at the fabulous Virgin Grand Resort on the island of St. John. The final four nights will find the group on the island of St. Croix.

Arrangements have been made to provide participants and their families with an exciting learning experience/vacation at the most affordable price. Please contact Dr. Dennis Harper at the address below for a complete set of brochures explaining the entire course/workshop and all the possible travel packages.

Dr. Dennis Harper
University of the Virgin Islands
St. Thomas, VI 00802 Phone (809) 775-1965

Logo: Search and Research

Research on Logo and Problem Solving

by Douglas H. Clements

We've discussed problem solving—what it is and how people do it. Now, the question is: Does Logo programming increase problem-solving ability? Unfortunately, the answer is: It depends on what we mean. We see students working on Logo problems. Most of us, however, would like to know if they're improving on their ability to attack these problems. Moreover, we'd like to know whether they get better at solving other problems. This is the issue of transfer.

Research on transfer falls into two categories—traditional, quantitative research and qualitative research. Studies in the later category are sometimes perceived as less scientific, but most agree that the narratives they generate are often intriguing and revealing. We'll look at each in turn.

Transfer: Some Good News and Some Bad News

Putting the computer aside for a moment, do students usually transfer what they have learned to new situations? Research on learning confirms our intuition—No, they don't. So it's no surprise that the news from traditional, quantitative studies of transfer from Logo to general problem-solving tests has ranged from good to neutral to bad.

On the positive side, some studies have hinted that Logo may increase general problem-solving ability. For example, Billings (1986) instructed fifth and sixth grade students in mathematical problem solving and heuristics using Logo. These students learned how to solve problems involving concepts and applications better than students who didn't experience this approach. Similar results have been reported in work with children from preschool to eighth grade (Grant, 1983; Lehrer, Harckham, & Archer, 1985; Littlefield, Delclos, Lever, Clayton, Bransford, & Frank,1988; Mann, 1986; Statz, 1974; Studyvin & Moninger, 1986). There is even some evidence that Logo programming can increase scores on standardized measures of intelligence (Chambers,1985; Clarke, 1985).

We can't draw any definite conclusions yet, though. As a counter example, Bruggeman (1986) found no effect of Logo work on fifth and sixth graders' ability to solve nonroutine, mathematical word problems. And again, several other studies report similar results—that is, they report no results—with both students described as normal (Fickel, 1986; Gaffney, 1985; LeWinter, 1986; Milojkovic, 1983) and special education (Horan, 1986) children.

Finally, compared to relatively direct training in problem solving, Logo may even give negative results. Dalton (1985) compared a seventh-grade Logo group to a problem-solving group specifically trained in the use of six problem-solving strategies. On a test including open-ended problems, the

Search and Research -- continued

problem-solving group performed better. These problems had no single solution. For example, "At the store plain beads are 3 for 10¢ and fancy beads are 5¢ each. If Bobby bought 20 beads, how many of each could he have purchased?" Here, any benefits of Logo did not transfer beyond the Logo environment. Targeting instruction directly at the desired behavior was more effective.

Worse yet, some say that Logo may teach incorrect processes. Seidman (1981) thought that fifth graders might learn problem-solving logic indirectly by using Logo's IF/THEN/ELSE conditional. He found no significant gains on tests of conditional reasoning abilities and warned that the Logo conditional may teach logical fallacies. Consider a pseudo–statement such as IF A THEN B. Knowing that A did not occur (or was not true), programming students were more likely to assume B did not occur. Although this makes sense in the programming context, it can lead to logical fallacies. If we say "if it is raining, then she takes her umbrella," we can not be certain that—just because it is not raining—she did not take her umbrella!

So Is There Any More Good News?

Even though the news is certainly not all bad, mixed results become discouraging after a while. However, there are reasons why they should not be. First, transfer is notoriously difficult to achieve with any educational intervention. So it may be unreasonable for us to expect consistent results when the art of teaching and learning Logo is in its infancy. It is also not reasonable to claim, as some critics have, that there is no transfer. If transfer did not exist at all, there would be no reason for education as we know it.

Second, there are qualitative reports of transfer of problem-solving abilities to non–Logo situations. A preschooler in a recent study we conducted was walking with his father on an ice and snow covered lake. He began walking in odd directions. In reply to his fathers inquiry regarding his behavior, he replied, "I'm writing my name." He continued, "Forward to here, a half circle—F, R, F, R, F, R, then L, L, then a slanted line. Uhhh...I can't put my pen up. I'll walk along the bottom." In this fashion, he traced out his name, ROB, in the snow. He told his father, "Even though I can't see it, I can picture the write commands in my mind."

More profound transfer may be possible, given the right conditions. In a longitudinal study, Lawler (1985) immersed his daughter in a Logo computer culture and observed her learning. This experience allowed her to be sensitive to instruction couched in the language of procedural programming. For example, Mariam used procedural terms learned in computer programming to help her father to jump rope. Speaking of the rope, she told him: "You hold it up when you're jumping...You [have] that pull–up bug." Mariam naturally used the language of procedures and debugging. Her sensitivity to procedural description transferred to school. When she reached first grade, her teacher reported that she enjoyed solving arithmetic problems and that she took more pleasure in the process than in merely getting the answer

There is even some evidence that, while programming, children frequently use the components we have discussed in earlier columns in *LX*.. Consider the following recorded episode from our research (Clements & Nastasi, 1988). Ken and Nick scrutinized a large circle on a computer screen—a circle that they had named BASEBALL. Nick, attempting to quickly finish the picture, tried to placate Ken: "It does look like a ball!" (an example of the solution monitoring metacomponent). Ken, not to be deterred until perfection was attained, protested, "But it doesn't look like a baseball. What's missing?" Nick asked urgently, "Do you think we could make the stitching?" (deciding on the nature of the problem). Ken concurred that the laces were the missing element and said, "Yeah! It would be like parts of a circle with little lines" (deciding on a representation). The pair began to decide on a strategy to create the arcs apportioned by short line segments to depict a baseball's stitches.

After several minutes, Ken cautioned Nick: "Before we put all these stitches in we'd better try to put the curve in the ball" (monitoring the solution and deciding on a new strategy.) With several moves, the boys positioned the arc in the appropriate place. "Look," said Nick, "it curves too much. When we put in the other curve, they'll overlap" (solution monitoring). In solving this new problem, the boys used the information acquired when creating the original circle for the baseball to determine the size of the needed arc. (This involved both monitoring and deciding on a strategy, as well as the use of knowledge-acquisitions components that we shall discuss in the future).

Where Do We Go From Here?

So children engaged in Logo projects may tackle significant, complex problems of their own design. They may use each metacomponent at each phase of completing their projects. But if so, what can we make of the bad news? That is, why are results from other studies so mixed? We need to dig deeper.

In the next few columns, we shall consider the metacomponents one at a time, to look more closely as specific problem-solving processes children use in Logo programming. This will also help us pinpoint the kinds of teaching strategies that might facilitate the development of each of these processes. Recall the four broad teaching phases that were presented in the previous column. They were worded in a general format because it is essential that they be used to help students solve both Logo and non-Logo problems. But it will be helpful know to flesh in specific details for teaching problem solving in the Logo context. We'll also do that in the next few columns.

Even now, however, there are teaching hints to glean from the students we reviewed.

Search and Research -- continued

- Never assume students are developing problem-solving abilities just because they are "doing Logo." Help students reflect on their problem-solving activities.
- If you have a specific goal, such as teaching certain heuristics, build those into your Logo program.
- As suggested in phase four, "Help students extend their abilities," encourage students to generalize the problem-solving processes they use across different settings. Use procedural language in non-computer settings whenever appropriate.
- At the same time, help them guard against overgeneralizations (e.g., logical fallacies).

Douglas H. Clements, SUNY at Buffalo
593 Baldy Hall, Buffalo, New York 14260
CIS: 76136,2027 BITNET: INSDH@UBVMSA

References

Billings, L. J., Jr. (1986). Development of mathematical task persistence and problem-solving ability in fifth and sixth grade students through the use of Logo and heuristic methodologies. *Dissertation Abstracts International, 47,* 2433A.

Bruggeman, J. G. (1986). The effects of modeling and inspection methods upon problem solving in a computer programming course. *Dissertation Abstracts International,* 47, 1821A. (University Microfilms No. DA8619363)

Chambers, S. (1985). *Children's acquisition of programming skills.* Manuscript submitted for publication.

Clarke, V. A. (1985). *The impact of computer on abilities and attitudes.* Manuscript submitted for publication.

Clements, D. H., & Nastasi, B. K. (1988). Social and cognitive interactions in educational computer environments. *American Educational Research Journal, 25,* 87-106.

Dalton, David W. (1985). *A comparison of the effects of Logo and problem-solving strategy instruction on learner achievement, attitude, and problem-solving skills.* Unpublished doctoral dissertation, University of Colorado, Boulder, Colorado.

Fickel, M. G. (1986). The effects of Logo study on problem-solving cognitive abilities of sixth-grade students. *Dissertation Abstracts International,* 47, 2066A. (University Microfilms No. DA8614452)

Gaffney, C. R. (1985). Computer languages: Tools for problem-solving? *Dissertation Abstracts International,* 45, 3097B. (University Microfilms No. DA8426831)

Grant, J., & Semmes, P. (1983). A rationale for LOGO for hearing-impaired preschoolers. *American Annals of the Deaf,* 128, 564-569.

Horan, R. E. (1986). Logo, A problem-solving tool. *Dissertation Abstracts International,* 47, 512A. (University Microfilms No. DA8602053)

Lawler, R. (1985). *Computer experience and cognitive development: A children's learning in a computer culture.* New York: Wiley.

Lehrer, R., Harckham, L. D., & Archer, P. (1985, April). *Microcomputer instruction in special education: First year findings.* Paper presented at the annual meeting of the American Educational Research Association, Chicago, IL.

(Continued on page 32.)

Global News

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

Our African correspondent, Seye Sylla, writes to tell us that she has taken a new position in Senegal which sees her in charge of a UNESCO regional project for the development of human resources. We wish her luck on this new endeavor and will be waiting to hear about some exciting African Logo happenings now that she has settled into her new job.

Another exciting Logo event will take place here in the Caribbean this summer in the form of a two-week workshop/course entitled Logo in Paradise. The workshop will be conducted by Tom Lough, Glen Bull and myself. It will take place from July 30 to August 12, 1989 on three of the Virgin Islands, St. Thomas, St. John and St. Croix in order for participants to experience all of America's paradise. The content of the workshop/course emphasizes the learning processes involved in the use of Logo in the schools. Topics will include LogoWriter, LEGO Logo, use of Logo with Apple Works and parallels between Logo and HyperCard.

Classroom applications discussed will include subjects such as:

- Use of Logo adventure stories in writing class
- Construction of Logo word machines
- Use of Logo in science and robotics
- Parallels between Logo functions and mathematical expressions
- Use of Logo with the physically handicapped

Graduate level credit from the University of the Virgin Islands will be available on an optional basis. The location of the course offers participants and their families an opportunity to see a unique part of the United States and a very attractive travel and accommodation package for participants and their families is available. Please write to my Virgin Islands address listed above for more information.

Our feature article this month reports on the Logo activities of the Cognitive Studies Laboratory in Porto Alegre Brazil where some very exciting and varied Logo projects are taking place. The following was submitted by Léa da Cruz Fagundes through our Latin American correspondent Jose Valente.

At LEC (Laboratorio de Estudos Cognitivos) in Porto Alegre, capital of the southernmost state of Brazil, we develop basic and educational research. Our aim is to build up knowledge in order to change educational practice, along with the development of a new culture with the computer revolution.

Global News -- continued

Based on Piagetian theory, we look for new ways of interpreting the learning process of children, adolescents, adults and the elderly. We use Logo programming activities in order to construct explicative patterns concerning information processing and knowledge representation.

We follow two basic lines. The first one develops studies on literacy, language arts, math, physics, art-education and music. The second one tries to build up interventional patterns in order to bring about changes in the learning environment of public school students. Our purpose is to study the influences this new learning environment has on the learning process. We also study the best way to prepare human resources in order to improve these processes and the changes they bring about.

The projects we are developing are:

Literacy: We started this project in 1984 with 3 students and now we have reached 160 students. Diverse experimental groups of children who have no preschooling are formed. The subjects are first graders, some of whom are repeating the grade one or two times. Results of the work of Emalia Ferreiro show that programming improves the conceptualization levels of the written word as well as writing ability.

Language Arts Our aims are to (1) investigate the possibility of using Logo as an aid for childrens' thinking and learning in natural language, (2) to comprehend how children understand and process linguistic information through the analysis of the way they build linguistic rules in Logo, and (3) to verify the effects of computer based environments over the child's linguistic competence. Twenty-four children are programming with words and lists.

Mathematics: We are studying childrens' spontaneous concepts of number, number relations, positional systems, non-Euclidean and Euclidean spaces, measurement, problem solving, and so on. All these ideas are developed and expanded within experimental activities in programming. Our studies intend to construct patterns of logical structures of the child's reasoning according to Piagetian concrete logic.

Physics: We are studying the cognitive mechanisms present in the physicogenesis of elementary physical notions during the Logo microworld-based activities of the experimental physics laboratory.

Art and Music Education Can the computer help develop the capacity to choose as well as the critical ability to translate ideas of the aesthetic consciousness? Can the Logo learning environment help the children to select and recreate these values? In order to answer these questions we are developing a research study with 10 students (10-12 years old) who are taking classes in a public art center.

Handicapped Children Different learning disabilities, cerebral palsied children, neurotic and psychotic children are being placed in Logo environments in order to attain a better diagnosis of their cognitive and affective structures.

Applied Research: Students enrolled in the Organizational Psychology courses are helping to design the Logo research in the schools. At these schools there are teachers who have been trained at LEC and work with children in Logo environments. Different training courses have been developed at LEC trying to reach the effective and efficient educational alternatives. Both classroom teachers as well as researchers at LEC are developing new tools and microworlds to enrich school life.

Presently, Logo studies have been conducted in six schools. 180 teachers have been trained, of which 71 are now in computer based environments. Since 1984 the program has involved 40 graduate students, 10 researchers and more than 3000 students of all ages.

For more information please write:

Léa de Cruz Fagundes (Chair)
Rua Sofia Veloso, 85
90050 Porto Alegre, RS BRASIL

(Continued from page 31)

LeWinter, B. (1986). A study of the influence of Logo on locus of control, attitudes toward mathematics, and problem–solving ability in children in grades 4, 5, 6. *Dissertation Abstracts International, 47*, 1640A. (University Microfilms No. DA8616959)

Littlefield, J., Delclos, V. R., Lever, S., Clayton, K. N., V. R., Bransford, J. D., & Franks, J. J. (1988). Learning Logo: Method of teaching, transfer of general skills, and attitudes toward school and computers. In R. Mayer (Ed.), *Teaching and learning computer programming: Multiple research perspectives* (pp.11-135). Hillsdale, NJ: Lawrence Erlbaum.

Mann, R. J. (1986). *The effects of LOGO computer programming on problem-solving abilities of eighth grade students.* Unpublished doctoral dissertation, United States International University.

Milojkovic, J. D. (1983). *Children learning computer programming: Cognitive and motivational consequences.* Unpublished doctoral dissertation, Stanford University.

Seidman, R. H. (1981, April). *The effects of learning a computer programming language on the logical reasoning of school children.* Paper presented at the Annual Meeting of the American Education Research Association, Los Angeles, CA.

Statz, J. (1974). The development of computer programming concepts and problem-solving abilities among ten–year–olds learning Logo. *Dissertation Abstracts International, 34*, 5418B-5419B. (University Microfilms No. 74-10,180)

Studyvin, D., & Moninger, M. (1986). *Logo as an enhancement to critical thinking*. Paper presented at the Logo 86 Conference, Cambridge, MA.

Logo Exchange

The Journal for Logo Activities Worldwide

Now published by the International Council for Computers in Education, *Logo Exchange* brings you exciting ideas from top Logo educators throughout the world.

Each of the nine issues includes:

- Logo articles *by* classroom teachers *for* classroom teachers;
- Columns which emphasize practical ideas for the use of Logo at the primary and intermediate level;
- Articles focusing on the use of Logo in language arts and math;
- Updates on Logo research;
- A look at Logo-like activities, software and hardware;
- Articles by Logo leaders from throughout the world;
- Reports on Logo activities world-wide.

The *Logo Exchange* is the journal for ICCE's Special Interest Group for Logo-using educators. SIGLogo members are invited to participate in local, regional, and national meetings and to contribute to the flow of ideas through the *Logo Exchange.*

Logo Exchange is published monthly September through May. SIGLogo membership is $24.95 for ICCE members (include membership number on order) $29.95 for non-members. Add $5.00 for non-U.S. memberships. All billed orders are charged $2.50 for handling.

To order, contact:
ICCE, University of Oregon, 1787 Agate St., Eugene, OR 97403-9905; ph. 503/686-4414.

Join SIGLogo before February 29th, 1989 and get the Logo Alphabet on disk FREE!

The Logo Alphabet is a set of procedures that draws all of the letters of the alphabet in the size of your choice. They were developed by Tom Lough, founding editor of the *Logo Exchange.*

You must use this order form to take advantage of the special offer to get the Logo Alphabet free when either **joining** SIGLogo (before February 29th, 1989) or **extending** your current membership for a year.

____*Yes,* I would like to join SIGLogo and receive the Logo Alphabet disk free of charge.
____*No,* I do not wish to join SIGLogo at this time, but I want the Logo Alphabet disk. I am enclosing $7.50.

Please check the version you want:

☐ Apple Logo (#LA1) ☐ Apple LogoWriter 1.1 (#LA3) ☐ Terrapin Logo (#LA5)
☐ Apple Logo II or LCSI Logo II (#LA2) ☐ Apple LogoWriter Version 2.0 (#LA4) ☐ Terrapin Logo Plus (#LA6)

	U.S.	Non-U.S.
SIGLogo Membership (includes the *Logo Exchange*)		
☐ ICCE member	$24.95	$29.95
☐ Non ICCE member	$29.95	$34.95
ICCE Membership (includes *The Computing Teacher*)	U.S.	Non-U.S.
☐ 1 year	$28.50	$31.00

Checks, VISA and Mastercard accepted. Add $18.00 per membership for AIRMAIL shipping of *Logo Exchange* outside the U. S. and Canada. Add $2.50 for processing if payment does not accompany your membership dues.

____Payment enclosed. Amount $_________(US Funds)

____ Charge: ___Visa ___Mastercard

Name on Card ____________________

Card # ________________ Exp. date _______

____Bill me. Add $2.50 for handling

Check one: ___ Renewal ___ New Membership

Name ____________________________

Address ____________________________

____

State _____ Zip/Postal code __________Country________

Phone____________ ICCE Membership #__ __ __ __ __ __ __

Membership # *must* be included to utilize ICCE member rates.

LONG DISTANCE LOGO

Educators—You don't have to go to classes to earn graduate credit—let the classes come to you! Introduction to Logo Using LogoWriter, a graduate level independent study course, allows you to learn at your own pace while corresponding with your instructor by mail.

WORK INDIVIDUALLY OR WITH A GROUP OF COLLEAGUES

Take Introduction to Logo Using LogoWriter at home, or study with a group of colleagues at school. The course uses a combination of video tapes (ON LOGO) featuring MIT's Seymour Papert, printed materials, textbooks, and diskettes. You view the tapes, read and report on course materials, do projects, design LogoWriter lessons for your students, and correspond with your instructor by mail.

SAVE $$$

If your district supports group training, *you* save money! Districts enrolling six or more educators in this course will receive a reduction in the fees of each person enrolled. To qualify, your district must provide lab facilities and a resource person with experience in computing to help answer questions.

NOT JUST ANOTHER CLASS

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

ON LOGO VIDEOTAPES

School districts may acquire a license for use of the ON LOGO package of eight half-hour videotapes and 240 pages of supporting print for $599. For a one-time fee of $1,295, the package may be obtained with both tape and print duplicating rights, enabling the district to build libraries at multiple sites.

Group Enrollment. A tuition of $260 per participant is available to institutions that enroll a group of six or more educators. This special price does not include the ON LOGO videotapes. Your group must acquire the tapes or have access to them. Once acquired, the library of tapes and materials may be used with new groups enrolling for the same reduced fee.

Individual Enrollment. Educators with access to the tapes may enroll individually for $290. Tuition including tape rental is $320. A materials fee of $60 per enrollee is charged for texts and a packet of articles. Enrollees who already have the texts do not need to order them.

Additional information and order blanks can be obtained from: LONG DISTANCE LEARNING, ICCE, University of Oregon, 1787 Agate St., Eugene, OR 97403-9905. Ph: 503/686-4414.