

THE NATIONAL LOGO EXCHANGE

Volume 4 Number 8

FORWARD 100!

April 1986 Page 1

Procedure Inputs: Local Variables or Not?

by Sharon Burrowes

Our introductory high school computer science course, a prerequisite for further courses in computing, touches on the usual "computer literacy" topics and teaches Logo programming. Last fall, I found myself with a large number of students who had never seen Logo, but who had been exposed to BASIC in junior high school.

In this lively group, there was a lot of bantering about which language was "best," usually in good fun. Since Logo has local names defined by using procedure inputs or the primitive LOCAL and BASIC does not, I decided not to teach or allow the use of MAKE (for defining global names) until after we had worked with procedure inputs for a while.

A Local Problem

Shortly after we had begun work with inputs, I posed a problem to help students discover how to use procedure inputs for sending values from one procedure to another. The problem went something like this: Define a START procedure which accepts two numbers as inputs and then uses those two numbers in two other procedures to produce a product and a quotient. Both computation procedures must be called from the first procedure.

The next day, I came to class armed with my solution, which looked like this.

```
TO START :NUMBER1 :NUMBER2
FIRST.COMPUTATION :NUMBER1 :NUMBER2
SECOND.COMPUTATION :NUMBER1 :NUMBER2
END
```

```
TO FIRST.COMPUTATION :FIRST :SECOND
PRINT :FIRST * :SECOND
END
```

```
TO SECOND.COMPUTATION :TOP :BOTTOM
PRINT :TOP / :BOTTOM
END
```

The Student Solution

The bell had barely rung when Brad gloated, "I figured it out!"

"Ha!" I thought, but I said, "OK, let's have it." He dictated the following solution which I entered on the demonstration system for the class to see.

```
TO START :NUMBER1 :NUMBER2
FIRST.COMPUTATION
SECOND.COMPUTATION
END
```

```
TO FIRST.COMPUTATION
PRINT :NUMBER1 * :NUMBER2
END
```

```
TO SECOND.COMPUTATION
PRINT :NUMBER1 / :NUMBER2
END
```

He didn't have inputs for either of the computation procedures. Once the procedures were defined, I self-assuredly typed START, fully expecting Logo to respond with

```
NUMBER1 HAS NO VALUE IN FIRST.COMPUTATION:
PRINT :NUMBER1 * :NUMBER2
```

Local Variables continued

Much to my surprise, the procedures worked! Doug was chuckling, Curtis was laughing, and Brad was having the time of his life! All my emphasis on the local nature of procedure inputs came crashing down!

Back to the Drawing Board

After class, I checked every Logo reference that I had. Everything told me that procedure inputs *were* private or local to the procedure in which they were defined. After a few moments at the keyboard, I concluded that the guys were right. But, at that point, I didn't understand *why* their procedures worked. However, I soon solved the "mystery," thanks to some knowledgeable friends.

I found that a procedure input is local to the procedure in which it is defined *and* to all the subprocedures called by that procedure, as long as the calling procedure is running.

This manner in which Logo uses local names (called "dynamic scoping") can be used to avoid passing values from one procedure to another using additional procedure inputs. However, it complicates debugging since subprocedures become dependent on their superprocedures. It is a poor programming technique to use, in part because the modularity of Logo's procedures is lost. You can no longer debug each subprocedure separately.

Once you move beyond elementary programming in Logo, the scheme which you use to define names becomes an important concern. Here is a series of examples using a combination of both global and local names to demonstrate the kind of problems that can develop if you use Logo names in the way just described.

To begin, type

MAKE "X 10

and then define this procedure:

```
TO PROCEDURE.1
PRINT :X
END
```

If you then type PROCEDURE.1, Logo prints 10, since 10 is a global variable value. Similarly, if you define another procedure which calls PROCEDURE.1,

```
TO PROCEDURE.2
PRINT :X
PROCEDURE.1
END
```

and then type PROCEDURE.2, the result will be 10 and 10, because the name X is global.

Change the Game

Next, make these modifications.

```
TO PROCEDURE.1 :X
PRINT :X
END
```

```
TO PROCEDURE.2
PRINT :X
PROCEDURE.1 :X
END
```

If you type PROCEDURE.1 15, then 15 is printed. If you type PROCEDURE.2, you may be surprised to find that 10 and 10 will be printed. Since no value is passed into PROCEDURE.2, it uses the *global* value of X (in this case, 10) and passes the 10 to PROCEDURE.1.

Now restore PROCEDURE.1 to its original form.

```
TO PROCEDURE.1
PRINT :X
END
```

If you type PROCEDURE.1, then 10 will be printed. Then modify PROCEDURE.2 as follows

```
TO PROCEDURE.2 :X
PRINT :X
PROCEDURE.1
END
```

and type PROCEDURE.2 20. Did you predict the result of 20 and 20?

This occurs because PROCEDURE.1 is a subprocedure of PROCEDURE.2 and PROCEDURE.2 has an input (which is local). This local name takes precedence over the global name X which has the value of 10.

The Local Wrapup

In summary, using the same name in many places can lead to confusion. Remember that:

1. Local names will be understood in all subprocedures of an active procedure.
2. The "most local" name always takes precedence. That is, if Logo encounters the same name, the value of the most local name will be used.

My experience with the scope of Logo names has made me more careful than ever about the scheme I use to create names in my procedures. It has also once again taught me the value of thoughtful and creative students. Who knows what they will teach me next?

Sharon Burrowes is the computer coordinator for the Wooster (OH) City Schools, and is currently serving as a member of the board of directors of the International Council for Computers in Education.

From The Editor

by Tom Lough

The Art of Intervention

Our son, Kyser, is well into his "terrible two's" now. But it is not all bad news. Like other youngsters his age, he is now beginning to attend to a task for more than fifteen seconds at a time! But, sometimes I'm afraid that I don't help his concentration very much.

Recently, he was involved with a jigsaw puzzle. I came into the room and immediately began to show him how to look for common colors in pieces that fit together. He would have none of it, of course! He had already established his own exploration routines. He didn't care if it took him longer to find the right pieces.

Later, I realized that I had intruded with no regard for what he was doing. My suggestion was made with no preparation or consideration of how my words would affect him. He resented my attempt to speed up his work. I resolved to do better next time.

The following week, he was working on the puzzle once more. This time, I approached quietly and spent a few moments observing what he was doing, trying to get into mental step with him. When I finally made a comment, it was with a better sense of timing and appropriateness. He smiled and we began to talk!

Still later, I thought about how I tried to apply the same idea to my students in their Logo work. Was I sensitive to what they were doing? Would I approach quietly and spend some time just observing? Did I try to get in step with them mentally before saying anything? Did I realize that sometimes it was not necessary to say anything? Would I restrain myself from giving solutions too readily in an attempt to speed up their work?

Effective intervention is an art. In order to be effective, intervention of any kind must be approached with sensitivity and empathetic understanding. I must always be aware of how my intervention will affect the thoughts, the mood, the process of my students.

This is not an easy task, and I certainly have not fully mastered it. But I find myself improving little by little. My students (and Kyser, too) seem to appreciate it when I intervene effectively.

What experiences have you had with intervention in your Logo work? I'd be interested in learning about them.

FD 100!



P.S.: Last month, we announced the availability of NLX brochures for distribution at conferences and in classes. We have filled many requests and even had to order a second printing! Thank you for your interest and support! Let us know if you need more!

Tips for Teachers

by Steve Tipps

Random Redux

RANDOM is a Logo function which has many entertaining and enlightening possibilities. From its use as an input, to games, to tool making in Logo, RANDOM can be used to make random things happen. Some possibilities and off-computer activities were explored in a previous column "Random Thoughts" (NLX, March 1984), but the time has come to look at random again.

Randomly, Randomly

The idea of the turtle being able to pick out any number from 0 to 50 is easy to show with a PRINT command in a REPEAT statement.

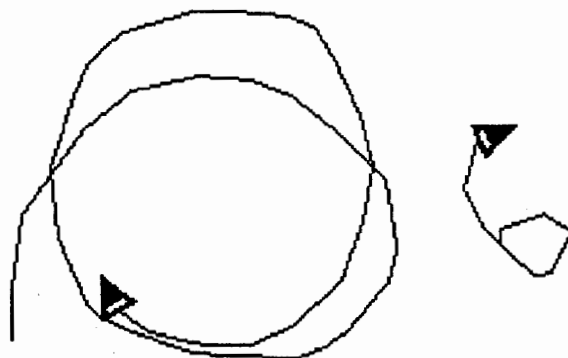
```
REPEAT 20 [ PRINT RANDOM 50]
```

```
21
17
44
...
```

Students can change the number 50 to another number and compare the highest value and the lowest value printed. Several trials should confirm the lowest and the highest results from an input value to RANDOM.

RANDOM can also be explored with distance and direction commands.

```
REPEAT 50 [ FD RANDOM 30 RT RANDOM 30]
REPEAT 10 [ FD RANDOM 20 RT RANDOM 90]
```



These two examples draw continually right turning figures. For one project, a student wanted a random turtle movement which was sometimes a little to the left and sometimes a little to the right. The purpose was to simulate a mouse running. This was an excellent problem with a "correct" programming solution using an offset which I was wise enough not to tell.

Instead, the student worked with small numbers hoping that the turns would not take the inevitable curve to the right. Then came a moment of inspiration.

Tips continued

The logic was explained to me like this:

"If RIGHT 20 LEFT 20 means the turtle is still going the same direction, then RIGHT RANDOM 20 LEFT RANDOM 20 would turn the turtle a little to the right then a little to the left. The difference between the turns would be the total turn depending on the size of the RANDOMs."

The invention led to a WOBBLE procedure which let the turtle move in a random, but approximately linear, fashion.

```
TO WOBBLE
LEFT RANDOM 20 RIGHT RANDOM 20
FORWARD RANDOM 20
END
```

Invention is wonderful learning, and this experience is one of many which alerts me to the value of Logo as a learning environment. I could have provided the approved answer with a long explanation about offsetting the values by adding 10 and subtracting random 20. However, the solution was just as effective and more intuitive.

Pickers and Choosers

Another use of RANDOM involves picking items from a list. The fundamental idea is to create a tool which can PICK any item from a list.

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

Understanding how PICK works may be interesting to some students, but the more important idea is to use PICK to do things. In their NLX column, "Listful Thinking," Glen Bull and Paula Cochran have used this procedure extensively in creating sentences and poems and stories. For example, PICK from lists of nouns and verbs for a sentence.

```
TO NOUN
OP [BOYS GIRLS MONKEYS TRUCKS TEACHERS]
END
```

```
TO VERB
OP [JUMP RUN TALK]
END
```

```
PRINT SE PICK NOUN PICK VERB
MONKEYS RUN
```

Other things can also be PICKed. A simple helper's list is one way to use PICK in the classroom. Use a procedure to output the list of students in the class.

```
TO STUDENT
OP [STACY TRACY JAMIE SHAWN KEVIN MELISSA]
END
```

```
PICK STUDENT
SHAWN
```

Random Fortunes

Another way to use PICK is as a fortune teller. Students are always interested in what will happen to them. Creating a fortune teller is a variation on the NOSY program shown in the January column. Students develop lists of possibilities to be used in a randomly generated future. Examples include future occupations, future places to live, or future transportation means.

```
TO OCCUPATION
OUTPUT [TEACHER SCIENTIST [MARINE BIOLOGIST] ]
END
```

```
TO PLACE
OUTPUT [VULCAN SEATTLE [ORION COLONY NINE] ]
END
```

```
TO TRANSPORT
OUTPUT [MASERATI ENTERPRISE PEQUOD]
END
```

```
TO TELL.FUTURE
PRINT SE [YOU WILL BE A] PICK OCCUPATION
PRINT SE [YOU WILL LIVE IN] PICK PLACE
PRINT SE [YOU WILL DRIVE A] PICK TRANSPORT
END
```

The resulting futures should be entertaining and lead to extensions of many kinds. Creating lists in procedures rather than as global variables make them easier to edit and eliminates many tacky issues of where to use quote and colons.

In Jeopardy

Another use of PICK would be to create games. An easy example is based on the television game show "Jeopardy." Each procedure can be a category with a list of answers.

```
TO PRESIDENTS
OUTPUT [WASHINGTON [16TH PRESIDENT]
[LEAGUE OF NATIONS] HERMITAGE [FOUR TERMS]
PEANUTS [MARRIED TO DOLLY] ]
END
```

```
TO STATES
OUTPUT [LARGEST [LAST TO BE ADMITTED
TO UNION] [EMPIRE STATE] [SHAPED LIKE MITTEN]
[EPCOT CENTER] [GRAND CANYON] ]
END
```

Tips continued**PICK PRESIDENTS
PEANUTS****PICK STATES
SHAPED LIKE MITTEN**

Categories can be invented to go along with any classroom units or interests. Of course, the inventors must keep track of the correct questions to go along with answers. A problem is likely to arise because the same item may be picked several times in a game.

Rather than solving the problem for students, let them come to some resolution. The programming solutions are more tedious than are the practical solutions.

Random Scheduling

Another use of PICK was suggested by Brenda Litwin, Logo enthusiast and junior high assistant principal in San Bruno, California. The problem she faced was to assign students to their science, English, and history classes. The students were already assigned to reading and math classes based on placement criterion. But she wanted the composition of the other classes to be as heterogeneous as possible. Her ingenious solution was to create a scheduling program in Logo to generate students' schedules.

The scheduler consists of procedures based on the assigned classes of students. If a student was already assigned to 2nd period math and 3rd period reading, then science, history, and English had to be scheduled for periods 1, 4, and 6. Based on this knowledge, a schedule called SCHEDULE.7C was written.

```
TO PERIOD.ONE
OUTPUT [SCI...JONES HIST...SMITH]
END
```

```
TO PERIOD.FOUR
OUTPUT [ENG...KEATS ENG...DUNNE ENG...HARDY]
END
```

Period Six has the same arrangement as Period One, but the schedule would be reversed. If a student gets history in the first period, science would be needed in period six. The PERIOD.SIX procedure takes care of the switch.

```
TO PERIOD.SIX
TEST :ONE = FIRST PERIOD.ONE
IFT [PRINT SE [6] LAST PERIOD.ONE]
IFF [PRINT SE [6] FIRST PERIOD.ONE]
END
```

Now the procedures are used to create a class schedule for someone.

```
TO SCHEDULE.7C :NAME
PRINT SE [SCHEDULE FOR] :NAME
MAKE "ONE PICK PERIOD.ONE
PRINT SE [1] :ONE
PRINT [2 MATH...GAUSS]
PRINT [3 READING...BOOKBINDER]
PRINT SE [4] PICK PERIOD.FOUR
PERIOD.SIX
END
```

This is an excellent example of using Logo to complete a task. The RANDOM operation is a very handy idea with can be used in many ways. Keep looking for new ways to RANDOM. Logo allows you to experiment with this extremely powerful idea.

If you have additional ideas to share, please send them to Steve Tipps, Midwestern State University, 3400 Taft Boulevard, Wichita Falls, TX 76308, or on Compu-Serve at 76606,1623.

Steve Tipps is the West Professor of Education at Midwestern State University in Wichita Falls, TX, and has been involved with Logo since 1982. He conducts Logo workshops for school systems throughout the United States, and is a popular conference speaker.

Logo Conference Update

Logo 86, the third international Logo conference in a series, will be held at the Massachusetts Institute of Technology, July 9 - 11, 1986. Preconference Logo tutorial sessions are scheduled for July 7 - 9. For more information, write to Joyce Tobias, Logo 86, MIT E15-309, 20 Ames Street, Cambridge, MA 02139.

The Educational Computer Consortium of Ohio (ECCO) announces its Spring Logo Conference on Friday, May 9, 1986, in Cleveland, OH. Seymour Papert is scheduled to be the keynote speaker, and a variety of Logo workshops and presentations are scheduled. For more information, write to Alice Fredman, Conference Director, ECCO Center, 1123 SOM Center Road, Cleveland, OH, or call (216) 461-0800.

Keene State College is sponsoring a two-day Logo conference as the culminating activity of a two-week Logo institute. Hosted by Dan and Molly Watt and Tony Stavely, the conference promises to be a lively affair. The program includes presentations by the institute faculty and participants, invited guests, and other Logo-using educators. A schedule of procedure plays, Logo music, and procedural contra dancing with live music will furnish many opportunities for informal learning. The conference is open to all. For more information, write to Tony Stavely, Keene State College, Keene, NH 03431, or call (603) 352-1909.

Logo Teacher Feature

by *Rebecca Poplin*

Featuring: Jim Fry

When most people think about their musical experiences, they have fond memories of groups gathered together singing or playing instruments. Music has a special ability to bring people together whether in work or play. Jim Fry, a Chapter I math teacher in Plymouth, Michigan, has been experiencing the pleasures of working with other teachers and students to create music with Logo.

Music Hath Charms

Jim's interest in Logo music was stimulated as he began using it to help his remedial math students understand mathematical relationships. He feels that the mathematics inherent in the patterns of music make a rich field for exploration with Logo. Jim and his students have written procedures to make musical phrases and music tools to transpose songs and reverse the order of the notes. In the process, the students and teachers decided to standardize the duration of the notes so that 32 is a whole note, 16 a half note, etc. What better way to help understand fractions?

This year, Jim has been working in a Logo integration project with a group of about 25 other teachers from several school districts in Oakland and Wayne Counties. He has been part of a sub-group working on music. With Austia Hollan and Lary Smith, Jim has completed a two part project, developing music games and music tools.

In the High - Low Game and the Long - Short Game, two students play at a time. One types in the command for notes and gives two numbers. The other student, not looking at the screen, tells which is high or low, long or short. In the Up - and - Down Game, the computer selects a pattern of five random notes, and students decide whether the pattern goes up or down. These games are designed for kindergarten through second grade.

From the Top

The second part of the project has been to create music tools for upper grade students. These tools are procedures but students use them as if they were commands. One procedure allows students to write a tune block. Other procedures transpose, mirror, and reverse the tune blocks created. Students can create a procedure for a whole song by making one or more tune blocks and manipulating the tune blocks with the other tools.

Other groups in the project are creating tools and activities to use Logo to teach skills in math and language arts. Jim's group has designed the procedures to work in Apple, Terrapin and Commodore versions. The Oakland Intermediate School District will soon publish the results of this project with documentation and disks. Ordering information will be published in NLX as soon as it is available.

All Together Now

When Jim first began using Logo, he felt isolated because he didn't know other teachers who were using the language. Jim's solution was to establish Logo Information For Teachers (LIFT), a clearinghouse group. Reggie Bowden and Anne Federlein worked with him to publish a directory of Logo using educators. More recently, the group has organized conferences at the University of Michigan focused on integrating Logo, not just teaching Logo as a programming language. Brian Silverman of LCSi and Tom Lough of NLX were guest speakers this year.

Jim and the LIFT group sponsor the "Teacher to Teacher" column in the NLX, and publish reviews of selected Logo publications.

Jim has also become involved in the CONFER Network set up through the University of Michigan. On the CONFER system, users participate in philosophical discussions as well as sharing practical teaching ideas. Anyone interested in this conferencing network may participate through Telenet or, in Michigan, through the Merit system. Further information may be obtained from:

Professional Development Office
University of Michigan
School of Education
Ann Arbor, Michigan 48103

Cooperation and sharing ideas are inextricably linked with Logo instruction. If you know of someone whose ideas and projects should be shared through this column, please contact:

Rebecca Poplin
2421 Fain Street
Wichita Falls, Texas 76308

Rebecca Poplin uses Logo to teach computing and mathematics at a junior high school in Wichita Falls, TX.

Krell Announces Logo Price Break

Educational institutions and school systems may purchase Krell Logo at \$14 per copy in lots of 100 or more. For more information, write to Krell Software Corp 1320 Stony Brook Road, Stony Brook, NY 11790, or call (800) 245-7355. In NY, call (516) 751-5139.

Logo Disserts: Dissertations Dealing with Logo

by Barbara Elias

A study by John Olive of Emory University may be of interest to high school mathematics teachers. Olive investigated the potential of Logo for developing students' reasoning skills in geometry in a learning environment supportive of relational learning.

Relational learning involves understandings which result from a learner's personal involvement with problem situations and ideas. It occurs when a learner is able to integrate and inter-relate information to solve problems. A synthesis of the *SOLO (Structure of Observed Learning Outcomes)* taxonomy and *Skemp* mathematical learning model provided the frame work for examining how students progressed through a series of instructional sequences in geometry. The taxonomy is a tool for helping teachers organize instruction in a hierarchy of levels of increasing abstractness.

Subjects for this study were 19 ninth grade students, randomly selected from a class of 39 students in a southeastern American community. The group consisted of 13 girls and 6 boys, ranging in ability level from "just below average" to "just above average." Most of the students had little previous experience with computers, and none had prior experience with Logo.

Using a guided discovery approach, students worked 3 days per week for six weeks on a series of Logo projects. They investigated simple regular polygons, variables in procedures, complex polygons, circles and arcs, and recursion. Rotations, reflections, and translations were also explored. A dribble file was made of each student's work. The files of a representative sample of 8 students were analyzed in detail using the *SOLO/Skemp* model.

Independent analyses of days 6 through 12, both by the researcher and by the evaluator taught by him, produced an inter-rated reliability correlation coefficient of 0.93.

Results indicate support of the *SOLO* taxonomy as a useful tool for studying learning processes. Some of the students were able to progress through the instructional sequences and demonstrate understanding of the Logo language and the geometric concepts (such as relational understanding) involved at the higher levels of the model.

Other students, however, showed confusion and regression with the fast pace and progressive complexity of the instruction. They were able to demonstrate only instrumental learning. (Instrumental learning generally involves rote memorization of information acquired from external sources.)

This suggests that instruction that is inappropriately paced may affect how students integrate new learnings. All students appeared to start each learning episode at the lower level and progress through successive levels. They tended to return to lower response levels when tasks became more complex.

Students who demonstrated relational learning, however, subsequently moved through the levels at a faster pace. These students also did more exploring in the early episodes. A number of helpful suggestions is included for workers interested in using dribble file data analysis.

>>>Olive, John (1985). *A Study of the Application of a Qualitative Taxonomic Synthesis to the Analysis of Geometric Reasoning in a CComputer Environment* (Doctoral dissertation), Emory University, Atlanta, GA 1985.

Many thanks to those who contributed dissertations and theses for this column during the year!

(Ed. note: Each month, Barbara Elias highlights a dissertation or thesis dealing with Logo. She is accepting copies of recent research results for the NLX Dissertation and Thesis Repository in preparation for the 1986-87 publishing year. Reports may be mailed to her in care of NLX, PO Box 5341, Charlottesville, VA 22905.)

Barbara Elias is an assistant professor in the Education Department of Virginia State University in Petersburg, VA, and a doctoral candidate at the University of VA.

NL-Textual Contest Winners Announced

Ricky Carter of Lesley College in Cambridge, MA, and Eric Nagler of Santa Clara, CA, have emerged as the winners of the NL-Textual Contest announced by Glen Bull and Paula Cochran in their January 1986 NLX column, "Listful Thinking."

The object of the contest was to create the best procedure or series of procedures which makes use of the fact that a procedure itself is a series of lists.

Both winning entrants wrote programs which print out a master procedure and all associated sub-procedures. Originally, there was only one prize announced. Since the two top entries were similar, an additional prize has been awarded.

Ricky Carter is the winner of a Koala Pad.

Eric Nagler wins a copy of Brian Harvey's book, *Computer Science Logo Style*. Eric's program not only prints out the procedures, but also puts a graphics dump on the same page.

Congratulations to Ricky and Eric! Thanks to all those who submitted entries. Thanks also to Glen and Paula for administering the contest. Details on other contests will be announced in future NLX issues.

Listful Thinking

by Glen Bull and Paula Cochran

Turtles and Lists: Never the Twain Shall Meet?

In the call for papers for the Logo 86 Conference, the program chair, Brian Harvey, raised several issues which he hopes will be addressed at the conference. One of these issues was raised by Uri Leron at last year's conference. He suggested that there are two Logos, the Logo of elementary turtle graphics and the Logo of advanced list processing, and that there is no easy way to get from one to the other.

Brian Harvey asks, "What about this? Is it true? Is it okay, or is it a problem? Is there, in fact, a reason to design one language for both ends of the spectrum?"

What's the Answer?

The answer is, "It depends." However, with the right approach, and a version of Logo which has the right features, we think that there is a natural transition from turtle graphics to list processing. In fact, in our introductory educational computing classes, we introduce teachers to lists and Logo poetry in the third week of the class. Turtles and lists become increasingly intertwined from that point to the end of the semester.

Papert talks about using familiar concepts to make new ideas easier to grasp. He calls this the "continuity principle". For example, children know how to move their bodies in space. This knowledge can be translated into a knowledge of how to move the turtle on the computer screen. Thus, Papert feels that turtle geometry provides a better introductory domain for learning mathematical thinking than does arithmetic. (Papert refers to learning something which isn't related to anything you already know as "disassociated learning.")

Turtle graphics is a good introduction to computer programming because it is concrete, and based on concepts which are already understood. It is easy to see what went wrong in a procedure when the turtle turns left instead of right, while the effects of a command which opens a file are less visible.

Prerequisites to Lists

It is desirable to have turtle graphics and list processing in the same language because it makes it easier to relate events in a list of words to parallel operations performed by the turtle. Thus, once turtle graphics becomes a familiar domain, it can be used as a stepping stone to lists. In other words, turtle graphics becomes a tool which is used to teach the concepts of list processing.

Two of the most important concepts introduced through turtle graphics are:

1. The concept of a procedure
2. The concept of modularity

Both of these concepts are crucial to advanced works with lists. The advantage of encapsulating a series of commands in a procedure is obvious with a shape such as SQUARE. (Well, sometimes it takes just a little coaxing, but everybody eventually gets the idea.) Modularity and the advantages of one procedure calling another become obvious with HOUSE.

```
TO HOUSE
SQUARE
FD 50 RT 90
TRIANGLE
END
```

A third concept which quickly becomes essential is the idea that information can be passed from one procedure to another. (Ed. note: See Sharon Burrowes' article on page 1 of this issue.) The power of passing parameters becomes obvious when one part of a HOUSE changes size (or doesn't, as the case may be).

As soon as Logo learners have had exposure to the concepts of procedures, modularity, and passing parameters, we go on to lists. Lists have gradually been introduced at an earlier and earlier point in the course. Three years ago lists were not introduced until the sixth or seventh week of the semester. Now they are introduced in the third week of the class.

By the third week, the educators in our course have already had some experience with lists. For example, they are already familiar with the PRINT statement:

```
?PRINT [ HI THERE! ]
```

The REPEAT statement also takes a lists of commands as an input.

```
?REPEAT 4 [FD 50 RT 90]
```


More extensive use of lists is introduced with the procedure PICK. As readers of this column know, this procedure randomly picks a word from a list of words.

?PRINT PICK [BREAD MILK EGGS BANANAS]
EGGS

It is a short step to creation of a procedure that contains a list of words, as a substitute for typing in the list each time. The advantages of not typing in a long list of words are parallel to the advantages of not typing in FD 50 RT 90 over and over.

TO ADJECTIVE
OUTPUT PICK [RAINY SUNNY WARM BLUSTERY]
END

Now the student can simply type:

?PRINT ADJECTIVE
WARM

We usually ignore the question of what OUTPUT is doing in the procedure, and so far no one has ever questioned its presence. When Paul Goldenberg publishes his book about Logo and language with its wonderful illustrations of input funnels and output spouts, then we will begin explaining it to beginners. Currently we defer this explanation until later in the semester.

An important point in this process is that we create this list as a procedure rather than as a variable. The reason for this is that, despite Brian Harvey's excellent explanations in *Computer Science Logo Style*, we have difficulty explaining the distinction between dots and quotes to beginners (and advanced users, for that matter). In fact, we find it considerably easier to teach the concept of lists than to teach the concept of variables. (We're familiar with the standard conventions for conveying the concept of a variable.)

As soon as we have ADJECTIVE, or some similar procedure which picks a word from a list of words, we have an excuse to introduce SENTENCE. For example,

?PRINT SENTENCE [SPRING IS] ADJECTIVE
SPRING IS BLUSTERY

The wonderful thing about introducing the SENTENCE operation this way is that it is resonant with what teachers already know about English sentences. In other words, the command makes sense in this context. Hal Abelson once lamented the use of English words with lists because of all the wonderful things which can be done with lists of numbers. However, once again, we're building a bridge. After a familiar context such as lists of English words is established, the concepts and mechanics of list processing can be extended to the less familiar domain of numbers.

Lists in Context: Language Arts

As soon as SENTENCE is introduced, we move on to Logo poetry, to provide a model of a content-related activity based on list processing. A POEM procedure can be constructed from several sentences, just as a HOUSE procedure can be constructed from several SHAPES. (You will have to create procedures for NOUN and PLACE. Perhaps you will want to expand the ADJECTIVE procedure, too.)

TO POEM
PRINT SENTENCE [SPRING IS] ADJECTIVE
PRINT SENTENCE [SPRING IS] ADJECTIVE
PRINT SENTENCE [SPRING IS A] NOUN
PRINT PLACE
END

It is important to establish that POEM is a procedure with all the attributes of a graphic procedure such as SQUARE. For example, just as it is possible to put SQUARE in a REPEAT statement,

?REPEAT 36 [SQUARE RT 10]

it is also possible to put POEM in a REPEAT statement.

?REPEAT 4 [POEM]

SPRING IS BLUSTERY
SPRING IS WARM
SPRING IS A BUTTERFLY
UNDER A BUSH

SPRING IS WET
SPRING IS WARM
SPRING IS A TOAD
IN MY POCKET

SPRING IS BRIGHT
SPRING IS FUN
SPRING IS A TULIP
IN THE GRASS

SPRING IS NEW
SPRING IS WARM
SPRING IS A RAINDROP
UNDER A LEAF

Lists without Turtles

With adults it is possible to go directly to lists and poetry without the intervening step of using the turtle as a vehicle for introducing the concept of a procedure. We prefer to use turtle graphics first to build a ramp. However, if we are only given two hours to demonstrate the use of Logo in the English curriculum, we might go directly to poetry or conversational dialogs in Logo.

Listful Thinking continued

We have found that, with novices, the minimum prerequisites for writing poetry programs in Logo are:

1. Ability to use the editor to alter a procedure
2. Ability to save work in a file on a disk

The ability to send work to the printer is also helpful. As later versions of Logo have made effective use of all four arrow keys for editing, it has been possible to teach a minimum subset of editing commands in shorter and shorter periods of time. It is much easier to use the arrow keys than to memorize a set of mnemonics associated with control keys. This seems like a small point, but beginners have so many new abstractions presented that this becomes important. We highly recommend more recent versions of Logo which take advantage of all the arrow keys for editing.

These prerequisites assume that novice teachers are given the following:

1. A poetry template or model to follow
2. The procedure PICK
3. Written documentation explaining how Logo poetry can be used in the curriculum
4. Hands-on assistance when help is needed

If these conditions are met, teachers with no Logo background can leave a morning workshop with a usable Logo activity for language arts. We do not recommend morning workshops as an ideal way to introduce Logo (particularly since it is possible to do more harm than good). However, it does demonstrate that, given a familiar domain of activity such as English, lists can be introduced in short order without undue fanfare or confusion.

When we describe POEM and this type of activity to people who are not familiar with Logo, they sometimes say, "Well, I could do that with Pascal (or C or FORTRAN ... or 8080 assembly language)."

We think that it is an instructive activity for them to write an equivalent sort of procedure in another language such as Pascal (we were just joking about 8080 assembly language). The issue is not whether it can be done, but rather the amount of overhead involved in introducing the activity to people who may never have used a computer before, and the ease with which this activity can be modified and extended to additional applications.

Changing Our Tune

Once lists have been introduced to users in the familiar context of English, they can be extended to other domains. We particularly like to use music as a way of extending lists to other areas. In the past, there often was not enough memory left in microcomputers to do

anything interesting after music procedures were loaded. Microcomputers now have more memory, and sometimes have special music chips in hardware. This has made it possible to write interesting music procedures which use lists of numbers.

In some versions of Logo, each number stands for a particular note, while on other computers or other versions of Logo, the number stands for a frequency that corresponds to a particular note. For example, the command NOTE [440] might produce the note A above middle C, which has a frequency of 440 hertz (cycles per second). A procedure could be constructed to play a list of notes in this fashion:

```
?PLAY [ 262 262 294 262 349 330 ]
```

This is the first line of "Happy Birthday", and could be put in a procedure:

```
TO LINE1
PLAY [ 262 262 294 262 349 330 ]
END
```

Of course, it is easy enough to write procedures which make use of conventional labels for notes rather than frequencies. However, if the objective is to get the student to use procedures that involve lists of numbers, it is sometimes better to wait until later before making this adaptation.

Music is a natural domain for programming because nearly everyone is familiar with music and notes, and because music tends to have a very modular structure. For example,

```
TO SONG
VERSE1 VERSE2 CHORUS
VERSE3 VERSE4 CHORUS
VERSE1 VERSE2 CHORUS
END
```

To date, we have avoided addressing music in this column because every version and dialect of Logo handles music in a different way. However, music is a very natural bridge to lists and numbers when it is available, and a great deal of experimentation can take place with respect to the numbers used and the pitch heard. A worthwhile project is to ask teachers (or their students) to write a procedure that produces the sound of a siren. How do the numbers of the musical frequencies relate to what we hear with our ears? (Ed. note: See "Teacher Feature" on page 6 for information on a Logo music project.)

From here it is possible to progress to other forms of mathematical activities involving lists. (There is a surprising amount of math in music alone, which may be the reason that, historically, there has been a natural affinity between mathematicians and music.) For example, by returning to turtle graphics, it is possible to

Listful Thinking continued

explore the patterns which can be created by interactions between lists of numbers and the movement of the turtle.

With a few judicious nudges from the teacher, it can be a short step to the use of lists with matrix algebra and other more advanced applications. For a plethora of ideas, we suggest David Thornburg's book, *Beyond Turtle Graphics: Further Explorations of Logo* (Addison - Wesley).

From Turtles to Lists

Is it necessary to have lists and turtle graphics in the same computer language? Obviously we think so. Without lists, the limits of turtle graphics are much more narrowly defined. Without turtle graphics, a concrete bridge to lists is missing. Each benefits from the presence of the other.

Is it true that there is no easy way to get from turtle graphics to lists? We don't believe this is true, if lists are introduced naturally and gradually, side-by-side with turtle graphics. We are quite ready to agree that if working with lists is deferred until all the concepts of turtle graphics are thoroughly explored, there may be trouble when the transition comes.

In teacher training, we introduce lists in the context of a content-related activity, such as POEM. We avoid discussing lists for the sake of lists (as we might in a computer science class). Consequently we do not have to sell teachers on the power of lists because they experience it firsthand.

Why are lists sometimes perceived as difficult? It is easy to forget how short a time Logo has been available on a widespread basis. It takes time to discover natural ways of introducing concepts. Teachers have had much longer to discover the best ways to introduce basic principles in physics and biology. Given a little time and experimentation, effective ways to introduce list processing in Logo will emerge.

Glen Bull is a professor in the University of Virginia's Curry School of Education, and teaches Logo courses at both the graduate and undergraduate level. His CompuServe ID is 72477,1637. Paula Cochran is finishing her doctorate at UVA's Curry School of Education. She is interested in Logo applications in language arts and speech-language pathology.

Logo Tour to Iceland and Holland

The International Logo Exchange announces a Logo tour to Iceland and Holland, August 10 - 23, 1986. Post-graduate credit is available. For more information, write to Educational Spectrum Inc., 188 Summerfield Street, Scarsdale, NY 10583, or call (914) 723-2242.

TurtleTips

by Jane Toth
and Donna Lanyi

Let your imagination soar! By this time of year, our students have become "seasoned Logo veterans," and are always amazing us with new ideas they discover. They especially like time-saving hints, such as the use of the SETPOS and SETHEADING commands.

The Turtle Teacher

As we've discovered, Logo is an excellent means of introducing and reinforcing geometric concepts to our students. They have already been exposed to arcs, circles, lines, and various shapes. Now comes the big challenge - coordinate geometry!

Most mathematics series do not tackle this concept until fifth or sixth grade. With Logo, a coordinate guide, and extra practice, younger students can grasp the concept.

We start off with our "handy dandy" graph paper. Students are shown the coordinates and then asked to number their papers correctly with an x- and y-axis. (Sometimes we laminate these papers for repeated use.) From the beginning, we plot the x-coordinate first, and then the y-coordinate.

Then, students use coordinates to find the starting points for various Logo drawings. For example, instead of using CS PU LT 90 FD 50 RT 90 to begin, they use CS PU SETPOS [-50 0].

On a large screen monitor, we mark points and ask students to estimate the coordinates. SETPOS is used to check the estimates. Once students understand coordinate points, we then demonstrate the usefulness of PRINT POS when trying to create a symmetrical shape.

When introducing the SETHEADING command, we always review the Total Turtle Trip Theorem, using a large laminated circle with degrees marked around the perimeter. Students quickly grasp the idea of turning the turtle to the exact heading rather than guessing the orientation with different intervals of turns. SETHEADING is particularly useful to students who have difficulties estimating turns.

A number of teachers have suggested that perhaps SETHEADING should be taught *before* the RIGHT and LEFT commands are introduced. If any NLX readers have experience with this idea, please share your thoughts with us.

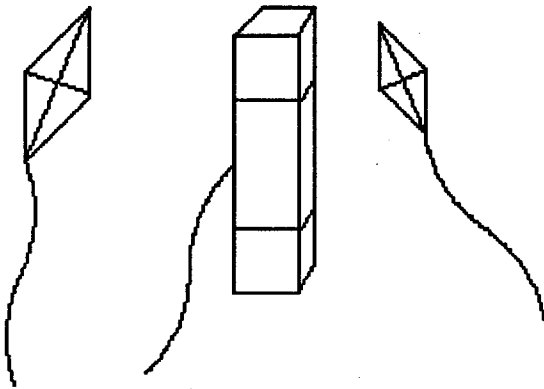
Procedure of the Month

Soar to new heights with Logo! While studying computers in today's technological world, students

TurtleTips continued

need to be reminded that the computer is often used for creating a first draft in the design and engineering trades.

Turn your classroom into a drafting room! What about a Logo kite project? Have students design their own personal kite on the computer. Then help them actually construct their kites. Set a date for "high flying," when the kites are sent aloft from the playground or parking lot. This project contains many related ideas for long-term projects.



If you are interested in a copy of our procedures for kites, please send us a stamped self-addressed envelope.

What interesting things are you doing with Logo as the school year comes to a close? Send your ideas to Jane Toth and Donna Lanyi, Beall Avenue School, 716 Beall Avenue, Wooster, OH 44691, or CompuServe 72477,2623.

Jane Toth and Donna Lanyi use Logo in their elementary classrooms in the Wooster (OH) City Schools.

NLXionary A Lectionary of Selected Logo Readings

*with Commentary and Opinion
by Griff Wigley*

"Logo, A Computer Language at the Crossroads," by David Thornburg, *A+ Magazine*, March 1986.

>David Thornburg is one of Logo's most prolific writers and is certainly a staunch Logo advocate. While he believes that Logo is primarily a language for learning, i. e., a tool for exploring content areas, his love of programming as a skill in and of itself and his knowledge of programming languages in general has always been evident in his writing.

>In this article, Thornburg bemoans the fact that Logo has not become more popular as a programming language. He cites several reasons for its failure to achieve greater respect among programmers:

1. the marketing of Logo as a graphics language,
2. its inexorable ties to Papert's educational philosophy,
3. its reputation for being slow and a memory hog, and
4. it has never been bundled free with computers as BASIC has.

>His hopes are on the rise, however, as Logo interpreters and compilers become available. He explains some of the spectacular results his Stanford graduate students in design were able to achieve using the ExperLogo compiler on the Macintosh, results that they had not been able to achieve using other languages. Some of the students' projects are now in the final stages of commercial development.

>He also contends that people who have become proficient Logo programmers have had a much easier time learning other languages such as LISP, PROLOG, Neon, and Forth.

>Thornburg mentions no ideas for solving this problem, and leaves many questions unanswered. Should schools start buying Macintosh computers instead of Apple IIs? Will the power of the new 16 bit Apple II provide the power needed? Should schools, in the name of programming, undertake the effort to change the emphasis from BASIC to Logo at the secondary level? Would emphasizing Logo's value as a programming language detract from its educational mission?

>If powerful versions of Logo complete with interpreters and compilers become available for the new 16 bit Apple II, then Logo may have a chance with secondary computer science departments. If college and university computer science departments begin using Logo routinely as their "foundation" programming language, and it replaces Pascal as the language for advanced placement, then it could be a shoo-in as the dominant all-purpose programming language.

>But don't bet on it in the near future. Logo's reputation as a language for little kids will not die easily, and that may be the greatest hindrance of all in the esoteric world of computer science.

If you would like to respond to this commentary, or know of other Logo related articles and issues which may be of interest to our NLX readers, please write to: Griff Wigley, 918 College Street, Northfield, MN 55057, or send electronic mail to me on CompuServe at 72007,24.

Griff Wigley is a facilitator in the Faribault (MN) Public Schools, and a school board member of Prairie Creek Community School.

LIFT's
Teacher to Teacher
 A Review of
New Logo Materials

by James Fry

This month, we look at a general Logo project book, and one with a much narrower focus.

Logo Projects for Everyone

Nudges: IBM Logo Projects, by Steve Tipps, Timothy Riordon, and Glen Bull, 1984, and *Nudges: Apple Logo Projects*, by Glen Bull, Timothy Riordon, and Steve Tipps, 1985, published by Holt, Rinehart and Winston, 383 Madison Avenue, New York, NY 10017.

Target Audience: All levels, from beginner to expert.

Content: The purpose of a "nudge," as the authors state, is to give "a little push in a direction." *Nudges* is a collection of 93 Logo activities, grouped into categories with such inviting names as "Here a Word, There a Number," "Turtle's Secret Codes," "Music Makers," and "What are the Odds?" They follow a common structure in that a little information is given to the student, followed by activities which help the student learn a new concept or skill. Then further explorations are presented and encouraged. Teachers and students are encouraged to take time to explore and discover as much as possible before going to another "nudge." An 84-page teacher's manual is also available, which contains information on the concepts presented, in addition to suggestions for more activities and explorations. The Apple book contains activities for both Apple Logo and Apple Logo II.

Format: *Nudges* is a paperbound book of over 270 pages, and includes a table of contents, an overview of beginning Logo commands, the 93 "nudges," and an appendix on file and workspace management. The IBM book also contains an index for both procedure names and for Logo commands. *Nudges* uses a helpful system of picture symbols (icons) throughout the book for preparation, caution, problems to solve, further explorations, cleaning up the workspace, saving procedures, and other activities.

Strengths: *Nudges* follows an excellent philosophy for working with Logo: give the students a "nudge" to get them started, but do not overstructure the activity so that it is difficult to explore. The effective use of icons is commendable.

Comments: I recommend that all Logo users and teachers of Logo "nudge" themselves to the nearest book store and pick up a copy of *Nudges*. It is an excellent addition to anyone's Logo library.

Logo in the Mathematics Classroom

Mathematics and Logo - A Turtle Trip Through Geometry, by Kathleen Martin and Donna Bearden, 1985, published by Reston Publishing Co., Reston, VA 22090, \$19.95.

Target Audience: Middle school students.

Content: The purpose of *Mathematics and Logo* is to help develop mathematics and geometry skills in the middle school. The book contains many activities and problems to allow students to explore and discover concepts involving angles, polygons, circles, and spatial relationships. The activities are described as if you were working with the students. Questions help guide the students to important mathematics ideas. The activities could be used with individuals, small groups, or classroom sized groups.

Format: The book is divided into a short introduction, 10 chapters each covering a different concept, appendices, and a list of resources. Appendix A shows many drawings which could be used with each chapter. Appendix B lists procedures in both LCSI and MIT Logo which could be possible solutions to the suggested problems in each chapter.

Strengths: The book is bound in a plastic spiral so it lies open. It could be used easily in a lab or classroom setting. *Mathematics and Logo* is an excellent resource book which shows how Logo can be integrated into the curriculum. The manner in which the book is written should help make readers feel more comfortable about using Logo. The many questions included should give a number of useful ideas to readers.

Comments: How about more books like this one?! For additional Logo and mathematics ideas, I recommend Martin and Bearden's *Microquests* publication. For more information, write Martin Bearden, Inc., 2502 Peach Tree, McKinney, TX 75069.

What new Logo materials and books do you find useful in teaching? We invite you to send information to the Logo Information For Teachers (LIFT) group, PO Box 5396, Plymouth, MI 48170, or send electronic mail on CompuServe to 76317,565.

James Fry uses Logo with his Chapter 1 remedial mathematics students at Novi Community School, Novi, MI, and is a co-founder of the Logo Information For Teachers (LIFT) group.

Q

and

A

by
Jim
McCauley

Q. I heard John Allen speak at a conference recently about something called "first-class objects." I asked around, but no one else seemed to know exactly what these objects are and how they relate to Logo. Can you help?

A. Dr. Allen is well known for his firm and somewhat controversial opinions on Logo, which are brilliantly presented in his book, *Thinking About [TLC] Logo* (Holt, Rinehart & Winston, 1984). If you are serious about

Logo, I urge you to obtain and study this book soon as possible.

Design of a Language

Computer languages have many components. Among these elements are variables, constants, literal data, functions, procedures, and data structures. In many languages, each kind of object has different uses and effects. Two important issues in designing computer languages are:

1. How many different types of objects should the language be able to work with, and
2. How should different types (or "classes") of objects be allowed to interact with one another?

A language that adheres to the principles of "first-classness" will have only a few types of objects and will allow them to interact freely with one another.

Languages tend to go to extremes with respect to these issues. In Pascal, for example, there are many different types of objects, and the ways in which they are allowed to interact are complicated and tightly controlled. The designers of Pascal feel that it is very important to keep control of these interactions. Without going deeply into the reasons, this can result in fast, efficient programs that can be carefully checked for errors.

Different Drummer

The Lisp / Logo community has a different orientation. There is only one data structure in Lisp: the list. Nearly all of the objects in a list processor can be used in the same way; that is, every object has a value associated with it, which it can supply when called upon. This allows the very simple syntax of Lisp to do very powerful things. Because any program-part can interact freely with any other program-part (an important element of first-classness), software tools developed in one

program can easily be transported to any other program and be expected to work perfectly well.

As for error-checking, the tradition of Lisp puts much of the responsibility for keeping programs bug-free in the hands of the programmer. Also, the power of the Lisp environment actually makes it reasonable to write programs that will check their own consistency and repair themselves when they go astray. Much of the work in artificial intelligence concentrates in this area.

Second Class Logo?

Logo retains much of the flavor of Lisp, although some implementations are closer to the ideal of first-classness than others. Most Logos are divided into commands and operations; this violates first-classness because the two types of objects don't behave in the same way.

Operations produce outputs, while commands do not. When one is doing simple things in Logo, this is not perceived as a shortcoming, but it looms as a more important issue when complicated projects are attempted. For example, if one attempts to add to Logo a new processing function that will accept both commands and operations as inputs, it is usually necessary to write two versions of the new function and add time-consuming code-checking features. If all versions of Logo adhered to the principle of first-classness (for example, by ensuring that both commands and operations produced outputs), this problem would not exist.

It is important to examine issues such as this which relate to the Logo environment. Thanks for your question.

.....

Each month, I will answer a selected question about Logo programming techniques or the thinking which leads to programming. If you have a question, send it to: Jim McCauley, NLX Q and A, 876 East 12th Avenue, #4, Eugene, OR 97401. Please enclose a stamped self-addressed envelope. You may also send questions to me at 70014,1136 on CompuServe's electronic mail system.

Jim McCauley is a graduate student at the University of Oregon, studying with David Moursund, and has written Logo articles for many national publications.

International Logo Exchange Published

If you are an educator who wants an international perspective on Logo activity worldwide, then consider a subscription to *The International Logo Exchange* (ILX) newsletter. This bimonthly publication contains reports from each continent as well as Logo conference and research information. \$19.50 per year, \$24.50 overseas. ILX, Box 5686, Charlottesville, VA 22905.

NLXual Challenges

by Robs Muir

Round and Round

As I write this, my wife and I are expecting our second child. Our situation has gotten me to thinking about circles lately. Life goes around in much the same way as our Logo turtle. We never go directly to our destination; every step causes us to head off in some new direction; and the final destiny could have been predicted from the start. But the journey is everything.

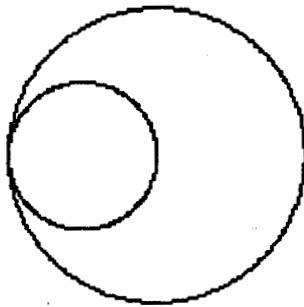
The Circle Primeval

It is interesting and instructive to watch children (and adults) drawing circles in Logo. There is a substantial number of Logo users who seemingly never learn any form of Logo circle other than the one they were first "taught." This usually looks something like

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

If they need a smaller circle, their solution might be

```
REPEAT 360 [FD 0.5 RT 1]
```



Never mind that the turtle draws the smaller circle in about the same amount of time. For some, the tyrant of Expedience reigns in (and reins in!) the land of Efficiency.

Circular Questions

As a teacher, I always ask two questions of my students as they begin to explore circles and the Total Turtle Trip Theorem:

1. Can you get the turtle to draw the same circles faster?
2. What are the fewest number of sides that you can use and still get an acceptable circle? That is, does REPEAT 15 [FD 10 RT 24] draw as aesthetic a circle as REPEAT 24 [FD 10 RT 15]? (Clearly, this is a function of

the resolution of the computer monitor, but the question, "What is the difference, if any, between a circle and a polygon?" cries out to be asked.)

Is there a "Platonic" circle unblemished in its purity, or are all circles simply "hi-res" polygons composed of very short, straight lines? Do the planets and their satellites revolve in circular orbits (ellipses, really), or does differential calculus help us to see the world as it really is - infinitesimal fractions of a second during which Newtonian physics demands that objects travel in straight lines?

What is a Circle, Anyhow?

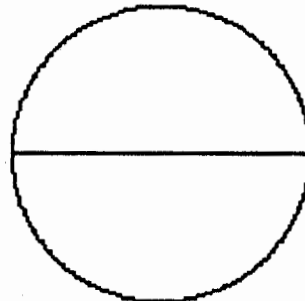
This month's challenge? Decide for yourself whether a circle is fundamentally different from a square. Then help me explore an interesting idea related to Steve Tipps' NLX column last month. (Ed. note: See "Tips for Teachers: Circular Quirks," NLX March 1986.)

If polygons are merely circles with fewer sides, is it correct to talk about the diameter of, say, a hexagon? Of course, RADIUS, CIRCUMFERENCE, and DIAMETER are mathematical words reserved for circles. But suspend your disbelief for a moment, and let's bend the English language a bit.

Here is a variation on POLY that insures that any POLYgon will have a horizontal line as its bisector, assuming that the turtle begins with HEADING = 0.

```
TO POLY :#SIDES :LENGTH
RT 360 / :#SIDES / 2
REPEAT :#SIDES [FD :LENGTH RT 360 / :#SIDES]
LT 360 / :#SIDES / 2
END
```

Do you see the logic in this POLY? You can draw a "circle" using POLY 360 1 and check its diameter by RT 90 FD (360/3.14159).

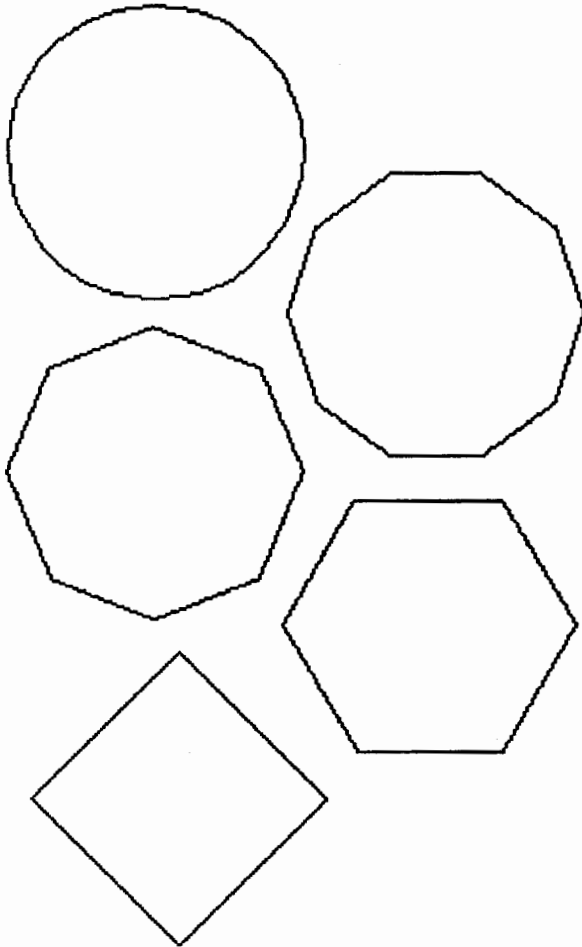


You can also approximate the diameter by simply fooling around with the turtle and carefully measuring the distances involved.

Question: How does this POLY differ from simply typing REPEAT 360 [FD 1 RT 1]?

NLXual Challenges continued

Here is a drawing of a 36-agon, a decagon, an octagon, a hexagon, a square, and a 2-agon (Are we "allowed" to talk about a two-sided polygon?).



In a "pure" circle, the circumference divided by π is said to be its diameter. Build POLYgons with a decreasingly smaller number of sides that all appear to have the same diameter, or in other words, can be inscribed in the 360-agon. Chart these characteristics about your polygons:

Number of Sides	Length of Side	Difference
10	35.5	4.5
9	39	
8	44	5
...	...	

As the length of the sides continues to increase, the difference between the lengths of the sides increases. Continue this until you have a one-sided(!) polygon. What happens to the differences along the way?

I Wonder...

The wonder of Logo can be found in how it can provoke us to ask questions we might not have asked ourselves before. Good Logo teachers provoke their students to ask the same unanswered questions they ask of themselves. And problem-solving is best learned with problems worth solving.

If you are a serious mathematician with the "definitive answer" to circles vs. polygons, or any other comments, we'd be happy to share your arguments with our readers. For a collection of such discussions, send a self-addressed stamped envelope to:

NLXual Challenges
 Attn: Robs Muir
 1688 Denver Avenue
 Claremont, CA 91711

You may also send other solutions and suggestions for additional challenges to me on CompuServe at 70357,3403.

Robs Muir is a teacher in the Claremont CA Unified School District and Claremont Graduate School. He and his wife, Candi, are celebrating the arrival of a son, Galen Andrew Muir, born February 24, 1986. Welcome, Little Turtle!

The National Logo Exchange, copyright © 1986 by Posy Publications, a part of The Posy Collection, all rights reserved. Published monthly, September through May, \$25 (US) per year mailed first class from Charlottesville, VA. \$5 (US) additional per year for addresses outside the United States, Canada, and Mexico. The opinions expressed by the authors are not necessarily those of The National Logo Exchange. Permission is granted for libraries and others registered with the Copyright Clearance Center (CCC) to photocopy articles herein for the flat fee of \$2 per copy of each article. Payment should be sent directly to CCC, 21 Congress Street, Salem, MA 01971. Address editorial and subscription related correspondence to: The National Logo Exchange, PO Box 5341, Charlottesville, VA 22905. Published since September 1982. ISSN 0734-1717. CompuServe 70020,223.

Editor.....Tom Lough