

THE NATIONAL LOGO EXCHANGE

Volume 3 Number 7

FORWARD 1001

March 1985

Up the Logo Ramp to List Operations

by
Tony Stavely

One of the things that charmed me when I first learned about Logo was the slogan, "No threshold, no ceiling." The more I thought about it, the more clear it seemed that the ideal learning situation would be just like a wedge. With no threshold, the learning begins immediately. An easy staircase would have fairly low risers and fairly wide treads. The limiting case is a ramp, an inclined plane gently rising forever.

However, learning Logo isn't exactly like strolling up an endless inclined plane. Exciting, frustrating sometimes, usually fun, but not easy. Furthermore, for many learners, passage into the domain of list processing means climbing a step with a pretty tall riser. What follows is some of my experience getting into that domain.

A FIRST STEP

My first challenge was to write a procedure to raise a number to a power. Terrapin Logo for the Apple has no such primitive, and I thought it would be a good first task after a dozen hours of making pleasing turtle designs. With the help of Bruce Metzger, a colleague who knew LISP, I eventually got the procedure TO RAISE :NUMBER :POWER so that it worked.

That project took more than a week and, although it did not contain list processing as such, it served as a kind of breakaway from turtle graphics. Gradually and steadily, with Bruce's help and, later, with the support of new friends Molly and Dan Watt and the Monadnock Logo Users Group that we founded, I kept posing little challenges to myself. Examples included finding the sum of a list of numbers, inserting an item at a specified location in a list, and creating a

game that learned the definitions of words from the user and asked about words it didn't know yet.

LOGO BREAKTHROUGH!

For me, the important breakthrough in mastering list processing happened while I was studying the humble LENGTH procedure in Harold Abelson's Logo for the Apple II (Byte/McGraw-Hill). LENGTH contained a key which helped me understand more of the power of lists. It turns out that whether you want to find the length of a list or the sum of a list of numbers, to reverse the order of letters in a word or words in a sentence, or do anything else with a set of Logo objects, there seems to be one fundamental format.

A typical procedure looks like this:

```
TO DISPLAY :ALIST
IF :ALIST = [] STOP
PRINT FIRST :ALIST
DISPLAY BUTFIRST :ALIST
END
```

This is the format:

- Part 1: procedure name
- Part 2: variable names
- Part 3: STOP rule
- Part 4: what to do with the FIRST of a list that is the current value of a local variable
- Part 5: recursive call, passing along the BUTFIRST of that local variable's list
- Part 6: END

Now study the ADDUP procedure listed below. ADDUP finds the sum of a list of numbers. Do you see the format in the procedure? Note the similarities between it and the DISPLAY procedure above.

```
TO ADDUP :ALIST
IF :ALIST = [] OUTPUT 0
OUTPUT (FIRST :ALIST) + ADDUP BUTFIRST
:ALIST
END
```

Logo Ramp continued

Unlike DISPLAY, ADDUP uses OUTPUT rather than STOP to halt things. This may be unfamiliar to some readers. All it means is that each ADDUP called upon gives (OUTPUTs) something to the procedure or command that calls upon it. The last ADDUP in the recursive sequence (the one that receives the BUTFIRST of a one-item list) OUTPUTs a zero. The next-to-last ADDUP OUTPUTs the sum of zero and the FIRST of its :ALIST (the last number of the original list). That sum is OUTPUT to the next-to-next-to-last ADDUP, to be added to the FIRST of its :ALIST, and so on, until the beginning ADDUP completes its addition task and OUTPUTs the complete sum to its caller.

ABOUT RECURSION

Recursion like this is a matter of knowing how to do just a few things and delegating everything you can't do to a helper. The helper turns out to be a copy of yourself. ADDUP knows the sum of an empty list (zero) and it knows how to add the first number of its list to the sum of the remaining numbers. However, it trusts its helper to take care of that part. While the helper is doing its job, the unfinished addition is stored in a "stack" inside Logo, waiting to be completed. If its helper calls a helper, the first helper's addition is stacked, too.

Once an empty list is reached, there are no more helpers needed; the unfinished arithmetic gets taken from the stack one number at a time. Readers may recall "stacking up" breakfast preparation while a family member cleans a skillet or runs next door for another couple of eggs. These recursive procedures do a similar thing. (Ed. Note: For additional OUTPUT ideas, see also "Q and A," NLX November 1984.)

TELL ME ABOUT IT

To share more aspects of Logo list processing and some of its uses in creating microworlds and personal tools, I am working on a book entitled Logo Tools: List Processing as a Powerful Resource. I would love to hear from NLX readers about experiences with list processing. What listful bugs creep into your work and build nests? What kinds of tasks would you like to do with Logo list processing? What projects do your students want to work on? With your assistance, Logo Tools may bridge the gap some learners find at the place where list work begins on the endless inclined plane of Logo.

Send your stories to Tony Stavelly, University of Michigan Professional Development Office, School of Education, Ann Arbor, MI 48104. After May 30, send them to me at Keene State College, Keene, NH 03431. Thanks!

 Tony Stavelly is a Professor of Psychology at Keene State College in New Hampshire, and a faculty member of The Logo Institute. He recently completed two years as co-editor of the NH Association for Computer Education newsletter and is completing a six-month stay at the University of Michigan.

From the Editor

Logo on the Move

Recently, I was privileged to present a Logo workshop to a very special group made up of high school physics teachers and university physics professors. Most sections of the United States and several parts of Canada were represented. Before things got going, I asked the participants to share a little information about themselves and what they knew or have heard about Logo.

To my surprise and delight, most of them reported that Logo was already being used in their high schools and universities! In fact, a number of these physics instructors were at the workshop because students were using Logo in math classes at their schools, and they wanted to be ready for them in physics.

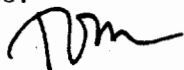
From all earlier indications, Logo has not made appreciable headway into the secondary and postsecondary fields to date. (See, for example, "Gradations of Logo," The Computing Teacher, December 1984 / January 1985.) Apparently, Logo's reputation as a child's language has made higher level educators reluctant to investigate its potential, in spite of early encouragement to do so. (See "Logo: Not Just for Kids," Harold Nelson, Microcomputing magazine, March 1982.)

I have always been concerned about this. A tool as powerful as Logo should not be neglected by the upper end of the educational community. Students at the higher levels are equipped with the thought processes necessary to tap the higher capabilities of the language. This is why I am particularly encouraged by the response of the physics teachers in my workshop. They helped me get rid of a case of the "Logo blahs."

I have used Logo with my college physics classes for nearly two years. It gives me a wonderful feeling to experience the excitement in the lab when someone makes an interesting discovery and everyone else rushes over to see. When I hear my physics students planning their next Logo experiment or help them set up a procedure to simulate some physical situation they thought up, I get a sense that something important is happening to them.

I wish for all higher level students this same feeling. Now it looks as if there might be a chance they can catch it.

FD 100!



Logo Announcements

Harvard Associates has a number of exciting Logo announcements! First, their PC Logo is now available on the Sanyo 550 Personal Computer series. A minimum of 128K of memory and one disk drive is required. The cost is \$149.95. Second, Harvard Associates has been selected to market the Valiant robot turtle, the first of a new breed. The Valiant has no cord connecting it to the computer! Instead, it uses an infrared communications link to pass computer commands. Thus, it can roam completely unencumbered, drawing complex designs or performing intricate choreography without your students having to worry about it getting tangled up in a cable. The Valiant is available for popular versions of Logo for the Apple II and IBM PC families, and the Commodore 64 computer. For more information, write: Harvard Associates, Inc., Attn: Valiant Turtle, 260 Beacon Street, Somerville, MA 02143, or call (617) 492-0660.

The Lisp Company offers versions of its TLC-Logo for several Z-80 based computers. Hailed as a second generation version, TLC-Logo possesses qualities of LISP and Smalltalk, yet is designed for a straightforward transition from the initial Logo versions in wide use today. TLC-Logo versions are available for the Epson QX-10, NCR Decision Mate 5, the NorthStar Advantage, and the Kaypro 284, 484, and 10. A Non-graphics customizable version is also marketed. For more information, write The Lisp Company, PO Box 487, Redwood Estates, CA 95044, or call (408) 354-3668.

Tipps for Teachers

by Steve Tipps

Survey Sense

The survey of surveys in last month's column suggested how Logo could be used as a data collection tool. For instance, information about birthdays or favorite pets might be collected with an interactive program with questions such as "Which month were you born?" One way to display the information in a kind of bar chart was also suggested.

Other ways of organizing, displaying and comparing information are also possible. Projects involving information in social studies, science, and mathematics from kindergarten through college level are easy to design. Tools which find high and low numbers, calculate the range, and compare results are useful. For tips on checking data as it is collected, see this month's Q and A column.

WHAT'S IN A LIST?

In the SURVEY procedure last month, information was collected as numbers for each month. You could check totals by printing out the names of the variables.

```
PO NAMES
"JAN IS 10
"FEB IS 8
...
"DEC IS 12
```

Or the value can be printed using PRINT and THING, a variable referent. In :JAN, the : is just short hand for the THING associated with the variable name JAN.

```
PRINT :JAN
10
PRINT THING "JAN
10
```

Working with the month names individually is not handy. A way to bind all the months together is needed. Here is an obvious place that a "list" of months is helpful.

```
MAKE "BIRTH.MONTHS [JAN FEB MAR APR MAY
JUN JUL AUG SEP OCT NOV DEC]
```

Tipps continued

Now you can ask questions about the list of months. What is the highest number of birthdays in any month? Which month has the most birthdays in this survey? How many responses were there altogether? What percent of birthdays were in each month? How does this survey compare to the survey last week? In each case, recursion and working with lists is used.

BIRTHDAYS MOST AND LEAST

Let's use a recursive procedure to make a chart about the number of birthdays in each month. The chart has five instructions, following the format shown in Tony Stavelly's article on page 1 of this issue.

If the list is empty, stop.

Otherwise, print the name of the first month.

Print some spaces, or tab over.

Print the value attached to the first month.

Take the first month off the list and perform the same actions.

Notice how the CHART procedure follows this plan closely.

```
TO CHART :LIST
IF :LIST = [] STOP
PRINT1 FIRST :LIST
TAB 4
PRINT THING FIRST :LIST
CHART BUTFIRST :LIST
END
```

```
TO TAB :NUMBER
REPEAT :NUMBER [ PRINT1 CHAR 32 ]
END
```

You might want to remember which month had the high number. The recursive plan works, but the middle of the FIND.MOST procedure is different. For each month, the number of birthdays is compared with whatever value is presently the highest.

```
TO FIND.MOST :LIST
IF :LIST = [] RESULTS.MOST STOP
TEST THING FIRST :LIST > :MOST
IFTRUE MAKE "MOST THING FIRST :LIST
IFTRUE MAKE "MOST.MONTH FIRST :LIST
FIND.MOST BUTFIRST :LIST
END
```

```
TO RESULTS.MOST
PRINT (SE [THE MOST BIRTHDAYS WERE IN]
:MOST.MONTH )
PRINT (SE [THERE WERE] :MOST [BIRTHDAYS
IN] :MOST.MONTH )
END
```

To start, you must first set the value of "MOST to 0 and the "MOST.MONTH to empty.

```
MAKE "MOST 0
MAKE "MOST.MONTH "
FIND.MOST "BIRTH.MONTHS
THE MOST BIRTHDAYS WERE IN APR
THERE WERE 13 BIRTHDAYS IN APR
```

The short variable name for the month winds up being a problem in the end.

Making a FIND.LEAST procedure is a simple variation. However, both procedures have not accounted for the possibility that two months might tie for high or low. Try writing a procedure for ties or pose it as a challenge for your students. Send me some of your solutions and ask for mine. Don't be hampered by the way I have organized the information. You may need to create other kinds of lists.

The high and low values are necessary to create another piece of information - the range. If the largest number of birthdays in a month was 13 and least number of birthdays was 6, the range would be 13 - 6 or 7.

```
TO RANGE
OUTPUT :MOST - :LEAST
END
```

```
PRINT RANGE
7
```

You also might write a procedure to correct the printed message in the "results" procedures if either :MOST or :LEAST equals 1.

You now have the beginning of a handy tool chest of procedures. The high, low, and range are very useful for describing many kinds of information.

CATEGORIES OF COUNTING

The information collected in the birthday survey is categorical data; the answers can be classified into groups. Although the months are in order, the order has no value associated with it. August is not a better month than July just because it comes after July. Even if we number the months 1 to 12, the order does not imply value. Other kinds

Tipps continued

of categorical data would be favorite names, pets, or ice cream. Each category is a collection of choices, not answers which have value such as grades of 70 or 90. Categories are analyzed by comparing the number of items in each category.

In a survey of ice cream favorites, six regular flavors and a seventh "other" category were used.

```
MAKE "FLAVORS [ VANILLA CHOCOLATE
STRAWBERRY PRALINE COCONUT MOCHA OTHER]
```

The results of the survey for Bowie School were charted and saved in a file.

```
CHART "FLAVORS
VANILLA 21
CHOCOLATE 14
STRAWBERRY 9
PRALINE 5
COCONUT 4
MOCHA 6
OTHER 21
```

```
ERASE PROCEDURES
SAVE "FLAVORS.AT.BOWIE
```

In Crockett School, the results were different and were saved in a different file.

```
CHART "FLAVORS
VANILLA 14
CHOCOLATE 7
STRAWBERRY 12
PRALINE 8
COCONUT 3
MOCHA 5
OTHER 11
```

```
ERASE PROCEDURES
SAVE "FLAVORS.AT.CROCKETT
```

One question might be whether the flavor favorites were similar or very different in the two schools. The numbers are certainly different, but that is partly because the number of students were different. Some way of comparing the results other than counting is needed.

PERHAPS PERCENTS

One way that categorical (or nominal) data is compared is with percentages. Percent is a statement of fractional, or proportional relationship. If you count 50 cars

and 15 of them are red, then 15/50 is a fraction which expresses the relationship of reds to the total. Other ways to write the fraction is .30 or 3/10. The percentage is

Logo can calculate the percentages. To get the percentage, the total of all the responses is needed. A recursive procedure is also used for the TOTAL. OUTPUT is used in TOTAL because the results are going to be used in another procedure. Note the similarity of this procedure to Tony Stavely's ADDUP on page 1. It does the same thing as TOTAL.

```
TO TOTAL :LIST
IF :LIST = [] OUTPUT :SUM
MAKE "SUM :SUM + THING FIRST :LIST
OUTPUT TOTAL BUTFIRST :LIST
END
```

Now you can print totals. Read in one of the data files from the ice cream flavor survey and calculate the total. Then read the other file and calculate. Be sure to set "SUM to zero each time you want the total.

```
READ "FLAVORS.AT.BOWIE
MAKE "SUM 0
PRINT TOTAL :FLAVORS
80
```

```
READ "FLAVORS.AT.CROCKETT
MAKE "SUM 0
PRINT TOTAL :FLAVORS
60
```

Dividing the number in each category by the total gives the fraction.

```
MAKE "SUM 0
PRINT (THING FIRST :FLAVORS) / (TOTAL
:FLAVORS)
```

Multiplying by 100 changes the decimal fraction into a percent. Then percent can then be rounded to a whole number with the ROUND operation.

```
MAKE "SUM 0
PRINT ROUND (THING FIRST :FLAVORS) /
(TOTAL :FLAVORS) * 100
```

This is the one operational line in a recursive routine which prints out the percentages of each category. A new variable "ALL was initialized so that calculating TOTAL many times was eliminated.

```
TO SET.VALUES :LIST
MAKE "SUM 0
MAKE "ALL TOTAL :LIST
END
```

Tips continued

```

TO PERCENTAGES :LIST
IF :LIST = [ ] STOP
PRINT1 FIRST :LIST
TAB 5
PRINT1 ROUND (THING FIRST :LIST) / :ALL
* 100
PERCENTAGES BUTFIRST :LIST
END

```

Percentages may be more useful in comparing the results of a survey than are the actual numbers because they adjust for different total numbers of responses. Be certain that you know which data file you are using.

```

READ "FLAVORS.AT.BOWIE
SET.VALUES :FLAVORS
PERCENTAGES :FLAVORS
VANILLA 26
CHOCOLATE 17
STRAWBERRY 11
PRALINE 6
COCONUT 5
MOCHA 7
OTHER 26

```

```

READ "FLAVORS.AT.CROCKETT
SET.VALUES :FLAVORS
PERCENTAGES :FLAVORS
VANILLA 23
CHOCOLATE 12
STRAWBERRY 20
PRALINE 13
COCONUT 5
MOCHA 8
OTHER 18

```

The percentages make it possible to compare the results. About the same percent of students at both school liked vanilla, coconut, and mocha. Chocolate was a favorite at Bowie, while strawberry and praline made a good showing at Crockett. Several more schools can also be surveyed and the results compared by keeping separate data files. Although the comparisons are only interocular (or "eyeballed") the sense gained through these comparisons is very important in reading and understanding many different tables and charts in every day reading. Working through your own surveys in Logo helps with your survey sense.

Steve Tipps is the West Professor of Education at Midwestern State University in Wichita Falls, TX. He presents Logo workshops for school systems throughout the United States.

NLXionary**A Lectionary and Discussion of Logo Readings**

by Griff Wigley

Time to liven things up, Logo lovers. If you haven't noticed, this column has been slowly changing over the last year. It started as sort of an annotated reader's guide to current and ancient Logo articles. As that became increasingly boring, I began editorializing more on each selection, first as to my judgement of its quality, and lately, using selections as vehicles for espousing my own perception of Logo reality.

Out of the goodness of my heart and the laziness of my bones, I want to afford NLX readers the same opportunity.

I am taking editor Lough's advice to stir the soup more. The column will become more interactive - sort of an electronic conference tree in print. I will sow the seeds of discussion by continuing to summarize current Logo articles. I will do my best to leave out my own editorializing until I have heard from those of you interested enough to comment on the articles. I especially encourage you to comment on the comments of others. Write to me at 918 College Street, Northfield, MN 55057. And now for this month's offerings.

"Logo Programming: Can It Change How Children Think?" by Douglas H. Clements, Electronic Learning, January 1985.

>Clements investigated the impact of Logo on children's cognitive abilities and cognitive development. Six pre- and post-tests were sandwiched around twelve weeks of Logo (two forty minute sessions per week) for a group of eighteen first graders, with a similar format for a control group of reading and math oriented computer assisted instruction (CAI).

>Results showed that, while there was no evidence of Logo affecting cognitive development any more than CAI, significant gains were indicated in the areas of problem solving abilities, creativity, and the abilities of the children to think about their own thinking, suggesting that Logo "may affect the way in which children use the cognitive abilities they possess."

NLXionary continued

"Logo Under Fire: A Conversation with Seymour Papert." by John Green, Classroom Computer Learning, January 1985.

>Just what the doctor ordered - an article touching on many of the controversies surrounding the educational use of computers, including Logo and artificial intelligence. Papert attacks the Bank Street study By Pea and Kurland, and yet is receptive to early results of a recent Canadian study by Higginson and Burnett, indicating a falling off of student competence with Logo in the second year. What might he say about the Clements study mentioned above?

"Is Computer Education Off Track? An Interview with Judah Schwartz." by Holly Brady and Melinda Levine, Classroom Computer Learning, February, 1985.

>Schwartz is co-director of the newly funded Educational Technology Center at Harvard. The Center's mission is to investigate ways in which various technologies can have a positive effect upon K - 12 math, science, and computer education. An advocate of the use of computers as tools, Schwartz is negative on teaching Logo in both elementary and high schools. He sees Logo as a "machine shop" in which any tool can be built, but that the work and skill to do so is unrealistic.

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

TurtleTips

by
Donna Lanyi
and Jane Toth

All of the "turtling around" during the school year certainly has produced some interesting graphic pictures. However, these terrific ideas are lost when the computer is turned off unless they are either saved as pictures or written and saved as procedures.

We have found that introducing the concept of procedure writing is fairly straightforward. The concept is capable of being handled by primary-age students. The difficulty comes when teaching the concept of editing. And, of course, editing is an integral part of procedure writing. Here are some thoughts for helping your students become Logo editors.

TEACHER HINTS

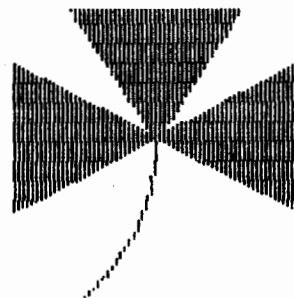
Students seem to be able to master editing fairly easily if only a few editing commands are presented at one time. Introduce only the necessary editing commands at first, and allow students to become familiar with the editing concept. As student editing skills are developed, introduce the more advanced editing commands and capabilities.

We post a list of these essential editing commands by each computer. Our students also have a personal list in their computer notebook. We have used a teacher prepared worksheet containing procedures with deliberate errors. Students are asked to define the procedures exactly as presented, including the errors, and then to use the editor to make corrections. (See also "Brookline Students Hunt Logo Bugs," by Jay Sugarman, NLX, December 1982.)

PROCEDURE OF THE MONTH

May you find your pot of gold at the end of the rainbow! For a St. Patrick's Day celebration, let your students design (and edit) procedures with an Irish theme - leprechauns, shamrocks, pots of gold, and rainbows. Our shamrock procedure can be used as a starting point for a St. Patrick's Day mural or bulletin board display.

Here is also an opportunity to make use of some of the modularity ideas you have already studied. For example, break the shamrock into parts and write a procedure for each part.



If you are interested in a listing of this procedure, please send us a self addressed stamped envelope. We invite any elementary teachers or students to send their procedures or ideas to be shared. Donna Lanyi and Jane Toth, Beall Avenue School, 716 Beall Avenue, Wooster, OH 44691.

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

Listful Thinking

by
Glen Bull and Paula Cochran

Powerful Selections

Last month we introduced the idea of selecting an item from a list. SELECT, you will remember, is a procedure that picks an item from a list, and then deletes the item. To use SELECT, first we have to make a list.

```
?MAKE "TOYS [BALL BEAR DRUM WAGON]
?PRINT :TOYS
BALL BEAR DRUM WAGON
```

Now we can SELECT a toy.

```
?PRINT SELECT "TOYS
DRUM
```

After the toy is selected, it is no longer in the list.

```
?PRINT :TOYS
BALL BEAR WAGON
```

Since SELECT deletes each choice from the list after selecting it, the same item is never chosen twice. This eventually destroys the list. For that reason, last month we also gave you a procedure for creating a list from the text of a procedure. The text is simply entered into a procedure, without PRINT or brackets.

```
TO ANIMAL.TEXT
RABBIT
FOX
DEER
SQUIRREL
POSSUM
CHIPMUNK
SKUNK
END
```

Then the CREATE procedure creates a list from the text.

```
?CREATE "ANIMAL.LIST "ANIMAL.TEXT
?PRINT :ANIMAL.LIST
[RABBIT] [FOX] [DEER] [SQUIRREL]
[POSSUM] [CHIPMUNK] [SKUNK]
```

There are two advantages to creating a list from text entered in the form of a Logo procedure (such as ANIMAL.TEXT):

(1) Large amounts of text are easier to enter using the Logo editor.

(2) It is easy to use CREATE to create the list a second time after all the items in the list are selected.

To see how this works, let's use SELECT with ANIMAL.LIST.

```
?PRINT SELECT "ANIMAL.LIST
DEER
?PRINT :ANIMAL.LIST
[RABBIT] [FOX] [SQUIRREL] [POSSUM]
[CHIPMUNK] [SKUNK]
```

DEER has been SELECTed and removed from the list. Let's try it again.

```
?PRINT SELECT "ANIMAL.LIST
CHIPMUNK
?PRINT :ANIMAL.LIST
[RABBIT] [FOX] [SQUIRREL] [POSSUM]
[SKUNK]
```

This time CHIPMUNK has been SELECTed and removed from the list. Eventually, SELECT will select and remove all the animals from the list. The list will remain empty until you use CREATE to create the list again for another activity.

Items in your list can consist of more than one word. Here is a text procedure that contains phrases.

```
TO PLACE.TEXT
BY THE POND
BETWEEN THE BUSHES
UNDER THE TREE
ON PINE NEEDLES
BENEATH WHITE CLOUDS
IN THE WATER
IN THE GRASS
END
```

Use CREATE to make a list of places.

```
?CREATE "PLACE.LIST "PLACE.TEXT
?PRINT :PLACE.LIST
[BY THE POND] [BETWEEN THE BUSHES]
[UNDER THE TREE] [ON PINE NEEDLES] [BENEATH
WHITE CLOUDS] [IN THE WATER] [IN THE GRASS]
```

Now SELECT can be used to select a place.

Listful Thinking continued

```
?PRINT SELECT "PLACE.LIST
ON PINE NEEDLES
```

The same strategy can be used to create a list of adjectives.

```
TO ADJECTIVE.TEXT
STILL
QUIET
NOISY
ALERT
QUICK
SILENT
YOUNG
END
```

```
?CREATE "ADJECTIVE.LIST "ADJECTIVE.TEXT
?PRINT :ADJECTIVE.LIST
[STILL] [QUIET] [NOISY] [ALERT] [QUICK]
[SILENT] [YOUNG]
```

THE VERSE THINGS IN LIFE ARE FREE

What can you do with lists using SELECT? One thing you can do with them is create free verse. With SELECT each verse will be different. Here is a short procedure that creates free verse about the woods.

```
TO WOODS.POEM
PRINT SELECT "ANIMAL.LIST
PRINT SELECT "ADJECTIVE.LIST
PRINT SELECT "PLACE.LIST
END
```

A verse is created each time you type WOODS.POEM.

```
?WOODS.POEM
DEER
QUIET
UNDER WHITE CLOUDS
```

```
?REPEAT-2 [ WOODS.POEM PRINT [ ] ]
SQUIRREL
QUICK
ON PINE NEEDLES
```

```
FOX
ALERT
IN THE GRASS
```

Eventually, all the elements in the lists will be used up. It simplifies matters to make a master procedure that creates all the lists again.

```
TO CREATE.LISTS
CREATE "ANIMAL.LIST "ANIMAL.TEXT
CREATE "PLACE.LIST "PLACE.TEXT
CREATE "ADJECTIVE.LIST "ADJECTIVE.TEXT
END
```

```
?CREATE.LISTS
```

The CREATE.LISTS procedure can be used by itself, or it can be placed in a procedure that writes verses.

```
TO FREE.VERSE
CREATE.LISTS
REPEAT 7 [ WOODS.POEM PRINT [ ] ]
END
```

This free verse program writes poems about the woods. Other free verse programs could write poems about the circus or the zoo, with different kinds of animals. Or, if the class is studying amphibians in science class, students could create poems about alligators and lizards and salamanders in English class.

ONCE UPON A TIME...

You can use the concept of SELECT to explore the idea of story sequencing. First create some text that conveys a story. This text is a story about an encounter with a mean dog. When you enter the text, be sure to press the RETURN key at the end of each line.

```
TO ENCOUNTER.TEXT
SALLY SAW THE MEAN DOG COMING.
SHE TURNED AND STARTED RUNNING AWAY.
THE DOG CHASED HER DOWN THE STREET.
END
```

After the text has been defined, it is necessary to use CREATE to create a list from the text before using it with SELECT.

```
?CREATE "ENCOUNTER.LIST "ENCOUNTER.TEXT
?PRINT SELECT "ENCOUNTER.LIST
SHE TURNED AND STARTED RUNNING AWAY.
```

```
?PRINT SELECT "ENCOUNTER.LIST
SALLY SAW THE MEAN DOG COMING.
```

In this instance, the events in the story are out of sequence. This is shown more clearly in a procedure that produces the entire story.

Listful Thinking continued

```
TO ENCOUNTER.STORY
CREATE "ENCOUNTER.LIST "ENCOUNTER.TEXT
REPEAT 3 [PRINT SELECT "ENCOUNTER.LIST]
PRINT [ ]
END
```

Here are some possible ENCOUNTER stories. The first story is clearly out of sequence. Sally starts running away before she sees the dog.

```
?ENCOUNTER.STORY
SHE TURNED AND STARTED RUNNING AWAY.
THE DOG CHASED HER DOWN THE STREET.
SALLY SAW THE MEAN DOG COMING.
```

The next story is more plausible, but still does not read as smoothly as it might.

```
?ENCOUNTER.STORY
SALLY SAW THE MEAN DOG COMING.
THE DOG CHASED HER DOWN THE STREET.
SHE TURNED AND STARTED RUNNING AWAY.
```

The final story is in the best order.

```
?ENCOUNTER.STORY
SALLY SAW THE MEAN DOG COMING.
SHE TURNED AND STARTED RUNNING AWAY.
THE DOG CHASED HER DOWN THE STREET.
```

This activity can be used as a springboard for discussing the importance of placing written information in the proper sequence. Stories the children themselves have written could easily be explored this way.

Another way to use SELECT is to create story starters. Of course, you don't have to limit yourself to just three sentences. You could create a list of a dozen sentences, and then SELECT three of them for each story starter. Or you could store different stories under different procedure names.

POWER TOOLS DISK OFFER

We have been placing various Logo versions of key procedures end of our column. For example, SELECT and CREATE were introduced last month, and are also included at the end of this column. (A slight improvement has been made in SELECT.) However, we recognize that some readers may not have the time to type in the procedures they need. Therefore, on a trial basis, we would like to offer this month's procedures on a disk. Send us a formatted blank disk and stamps or international coupons for return postage. This offer expires March 31, 1985.

NLX March Story Tools
Curry School of Education
University of Virginia
Ruffner Hall, 405 Emmet Street
Charlottesville, VA 22903

When you send your blank disk, be sure to indicate which version of the procedures you want:

- (1) IBM Logo
- (2) Apple Logo
- (3) Apple Logo II
- (4) Terrapin 1.0 versions
- (5) Terrapin 2.0
- (6) Krell Logo
- (7) Commodore Logo

PROCEDURES

PICK and CREATE are the same for all Logo versions listed above.

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

```
TO CREATE :LIST.NAME :PROCEDURE
MAKE :LIST.NAME BF TEXT :PROCEDURE
END
```

Here are the remaining procedures for IBM Logo and Apple Logo.

```
TO SELECT :LIST
IF THING :LIST = [ ] [OUTPUT [ ] ]
MAKE "ITEM PICK THING :LIST
DELETE :ITEM :LIST
OUTPUT :ITEM
END
```

```
TO DELETE :ITEM :LIST
MAKE :LIST DELETE.LOOP :ITEM THING :LIST [ ]
END
```

```
TO DELETE.LOOP :ITEM :OLDLIST :NEWLIST
TEST EMPTY :OLDLIST
IFTRUE [PRINT SE :ITEM [NOT FOUND]]
IFTRUE [OUTPUT :NEWLIST]
TEST :ITEM = FIRST :OLDLIST
IFTRUE [OP SE :NEWLIST BF :OLDLIST]
OP DELETE.LOOP :ITEM BF :OLDLIST LPUT FIRST
:OLDLIST :NEWLIST
END
```

Listful Thinking continued

Here are the remaining procedures for Terrapin Logo on the Apple and Commodore 64 computers.

```
TO SELECT :LIST
IF THING :LIST = [] OUTPUT []
MAKE "ITEM PICK THING :LIST
DELETE :ITEM :LIST
OUTPUT :ITEM
END
```

```
TO DELETE :ITEM :LIST
MAKE :LIST DELETE.LOOP :ITEM THING :LIST []
END
```

```
TO DELETE.LOOP :ITEM :OLDLIST :NEWLIST
TEST :OLDLIST = []
IFTRUE PRINT SE :ITEM [NOT FOUND]
IFTRUE OUTPUT :NEWLIST
TEST :ITEM = FIRST :OLDLIST
IFTRUE OP SE :NEWLIST BF :OLDLIST
OP DELETE.LOOP :ITEM BF :OLDLIST LPUT FIRST
:OLDLIST :NEWLIST
END
```

In Terrapin Version 1, PICK also requires these procedures:

```
TO ITEM :NUMBER :LIST
OUTPUT ITEM.LOOP :NUMBER :LIST 1
END
```

```
TO ITEM.LOOP :NUMBER :LIST :COUNT
IF :LIST = [] OP (SE [THERE ARE ONLY] :COUNT
[ITEMS IN THIS LIST.] )
IF :NUMBER = :COUNT OP FIRST :LIST
OP ITEM.LOOP :NUMBER BF :LIST :COUNT + 1
END
```

```
TO COUNT :LIST
OUTPUT COUNT.LOOP :LIST 0
END
```

```
TO COUNT.LOOP :LIST :NUMBER
IF :LIST = [] OUTPUT :NUMBER
OP COUNT.LOOP BF :LIST :NUMBER + 1
END
```

 Glen Bull is a professor in the University of Virginia's School of Education, and teaches Logo courses at both the graduate and undergraduate level. Paula Cochran is a reformed English major who studied linguistics at the University of Cambridge. She is now a speech-language pathologist working with language-disabled children.

Logo Disserts:

Dissertations Dealing with Logo

by Barbara Elias

Can an inservice training program serve as an effective means for preparing teachers to use Logo as well as assist them in designing and using a Logo curriculum? This question was investigated in an exploratory study involving eleven elementary and middle school teachers. Previous microcomputer experience ranged from little or none to a maximum of two computer education courses.

A one week intensive workshop on the Logo language itself and how to teach a Logo centered curriculum was used for the first phase of teacher training. Workshop sessions included experiences with modular programming, iteration, variability, and planning.

The implementation phase included follow-up weekly observation and feedback conferences ("coaching") with each teacher. This phase also incorporated teacher learning team sessions for sharing experiences and providing additional support. A variety of materials were provided for the teachers.

Teacher interviews, self report records, and Logo skills test scores provided data for the study. Results indicated that most teachers developed a measure of expertise with Logo and success with the Logo curriculum. The results also lend support to the view that teachers who are involved in introducing Logo and Logo curriculum materials need training as well as follow-up support.

>>>Ferres, G. W. (1984). Training and Implementation Strategies Appropriate to the Introduction of Logo into Teachers' Curriculum and Instruction. (Doctoral dissertation, University of Oregon, 1983). Dissertation Abstracts International, 44, 3264-A.

(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. 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 Virginia.

Teacher to Teacher

Teacher Training Materials with Classroom Applications

reviewed by
Anne Cairns Federlein

Resolutions! At midnight, December 31, 1984, the co-editors of this column resolved, among other things, to continue the quest to find exciting Logo classroom materials and teachers who are using them. This is the only resolution which my three colleagues have allowed me to divulge publicly. But I can tell you this ... it is the only resolution they have kept!

ANOTHER MECC FIND!

Teachers new to Logo will find Introduction to Logo for Teachers Training Materials by Marcia Horn of the Minnesota Educational Computing Corporation (MECC) an answer to their innermost fears! Although designed to be used in a Logo teacher training course, the book is a most worthwhile addition to any Logo classroom as well. Let me tell you why.

The primary emphasis of the book is to help teachers emphasize the problem solving process with Logo, not necessarily concentrating on the language itself.

Modules for Apple Logo divide the book, beginning with "Meeting the Turtle," and "Discovering Patterns," then progressing to "Teaching Turtles," "Using the Editor," "Changing Pen and Background Colors," and "Creating Building Blocks." More difficult lessons include "Procedures," "Recursion," and "Words and Lists."

The transparency masters in Section II enrich the lessons with good visual aids, and are accompanied by suggestions on how to develop a Logo journal, reinforce new vocabulary, and guide groups on working effectively.

The activity sheet section contains projects that are as open ended as we have reviewed to date. For example, one sheet suggests a format for a Logo journal, with statements such as "What I want to do," "What I did," and "What I learned," giving guided structure and valuing the child's responses, yet avoiding rote "right" answers. Another encourages a team of children to record the steps in their thinking as they design procedures.

HELP! HELP!

The reference section is priceless to the teacher struggling with Logo and computer operations. Apple Logo short word cards, editor cards, a Logo command summary, and diskettes for both students and teacher courses fill immediate needs. Long term needs are addressed by references to books, other classroom materials, magazines, user groups, newsletters, and the MECC hotline number.

This book is highly recommended for the beginning teacher in Logo. I wish it could be included in every educational sale of an Apple II computer!

Introduction to Logo for Teachers Teaching Materials is available for \$49 from MECC, 3490 Lexington Avenue North, St. Paul, MN 55112, or call (612) 481-3527.

We welcome reader comments and opinions on this review. Send your comments to Logo Information For Teachers (LIFT), PO Box 5396, Plymouth, MI 48170.

Anne Cairns Federlein is a professor in the Early Childhood Area, School of Human and Educational Services, Oakland University, Rochester, MI.

Logo Notes

Logo Computer Systems Incorporated (LCSI) announces a price reduction of \$100 for its Sprite Logo for the Apple. The new price is now \$199 per copy, and includes a plug-in Sprite board, language disk, and three manuals. Quantity discounts are available. For more information, write LCSI, 555 West 57th Street, Suite 1236, New York, NY 10019, or call (212) 765-4780.

Terrapin, Inc., has available a networked version of its popular Logo version for the Apple II personal computer family. Configured for the Corvus hard disk system, which allows up to 64 computers to be interconnected, the networked Logo provides a flexibility not available to teachers using individual computers. For more information, write Terrapin, Inc., 222 Third Street, Cambridge, MA 02142, or call (617) 492-8816.

K-12 Micromedia offers a free 24-page catalog of Logo books, teaching aids, and utilities. This is the only Logo collection of its kind, and contains Logo goodies for the novice and expert alike. To order a copy, write to K-12 Micromedia, 172 Broadway, Woodcliff Lake, NJ 07675.

前 100

by
Hillel
Weintraub

Many Japanese companies recognize the potential of the educational market for software and computer products. Drill and practice materials are in demand by the "juku" (tutoring schools where students study after regular school hours.). Several companies are working to develop computer materials for these numerous private schools.

Personally, I have great doubts about the overall effectiveness of this type of education. It seems to be an example of the dissociated learning to which Papert referred in Mindstorms. Yet, the Japanese children seem to develop a certain kind of powerful involvement due to the competitive atmosphere of these schools and of the educational system in general. Winning this memorize-it-spit-it-out-on-the-test game is more important than learning to most students. (Ed. Note: For more information, see "Japanese Education and Its Implications for US Education," by Nobuo K. Shimahara, Kappan, February 1985, and "Of Sushi and Mushi," by Duane Yee, NLX, January 1984))

ONE COMPANY STEPS OUT

I'd like to tell you about one innovative company which has taken a rather daring step into educational areas. Three years ago, Uny Company (one of Japan's largest retail organizations) formed its BYNAS division, devoted to developing and providing application systems, and acting as a liaison between users, scholars, educators, and manufacturers. As part of their educational computing interests, they have become a leader in Logo work in Japan. But BYNAS was not established with the expectation of immediate results.

Beginning as the Japanese representative for Terrapin Logo, BYNAS is now hoping to represent all Logo versions equally, including any they develop. This seems to put them in a rather unusual position of representing competing products. However, their intent is the growth of Logo-like education rather than the growth of a particular brand of Logo. More and more companies are showing a willingness to cooperate with them as this intention becomes more clear.

LOGO JOURNAL

An especially interesting BYNAS development is the publication of Logo Journal, which began last February. This Japanese language journal is making a serious effort to share information on all Logo versions as well as keep its readers up to date on new educational and commercial Logo happenings. But it is difficult for a company publication to gain the status and respect normally reserved for a university scholarly journal or other independent publication.

As a contributing editor, I write occasional articles for them or help with some English expressions in translations into Japanese.

The BYNAS staff members keep abreast of foreign published Logo materials and Logo developments in Japan. Agreements with both the NLX and YPLA newsletters and The Computing Teacher magazine allow for the translation and reprinting of articles of interest to the Japanese Logo community.

All articles are in Japanese, but most procedures are listed in English. With a circulation of about 350, this 16-page publication is available by subscription for \$50 per year, air mail postpaid. For more information, write Uny Co., BYNAS Division, 3rd Floor, Dainagoya Bldg., 28-12 3-Chome, Nakamura-ku, Nagoya, Japan 450.

Until next time, MAE 100!

Hillel Weintraub is a teacher at Doshisha International High School in Kyoto-fu, Japan, and is the president of the Society for Microcomputing in Life and Education (SMILE).

Logo Notes

Are you a member of a Logo user group? If so, make sure your group is included in the first edition of The National Logo Exchange's Logo User Group Directory. Scheduled for publication this summer, the directory is planned to help establish communications between user groups, and to provide information to prospective members. A courtesy copy of the directory is sent to each participating user group. Send the name and address of your group, the name and telephone number of a contact person, the size of your membership, your objectives and interests, and a schedule of your meetings to NLX User Group Directory, PO Box 5341, Charlottesville, VA 22905.

Q and A

by
Jim McCauley

Q: I've been trying to learn how to trap errors in user input in Logo programs. For example, with this piece of Logo code,

```
PRINT [GIVE ME A NUMBER AND I'LL ADD 20
TO IT.]
MAKE "NUM FIRST READLIST
PRINT :NUM + 20
```

If someone types in something other than a number (like the word FRED, say), the program dies because "+ DOESN'T LIKE FRED AS INPUT" or some such reason. I know how to do this in BASIC using loops and IF statements, but doing it in Apple Logo seems to involve the use of the commands CATCH and THROW, which I find really hard to figure out. Can you help?

<[-?->

A. Yes, I'll try. But let me suggest an approach other than CATCH and THROW, which are very complicated indeed and which do not appear in all Logo dialects.

The idea of error-trapping is to make sure that incorrect types and forms of data do not sneak into programs and bomb them in the way you describe. In languages like BASIC, this information enters the program through something like an INPUT statement.

```
100 PRINT "GIVE ME A NUMBER AND I'LL
ADD 20 TO IT."
110 INPUT N
120 PRINT N + 20
```

Note the similarities to your Logo code fragment. If this BASIC program is run and the user types in something besides a number (such as FRED) instead of a number, the program will choke. To make sure this doesn't happen involves checking the type of input before it is passed on to other parts of the program.

To guard against repeated errors in input, a checking routine is usually placed in a loop from which there is no escape until the type and form of the input are correct.

While it is possible to create similar checking loops in Logo, there is a very different way of thinking about this kind of

problem. Consider this: Logo programs can be thought of as "strong" when they "take care of themselves" with internal methods of guarding against errors committed by users. In Logo, we program in terms of procedures. How can we create procedures that are "strong?"

LOGO FILTER

One way would be to develop a "functional filter," an operation that captures incoming data and inspects it. Data of the correct type and form are allowed to pass through, while erroneous input is trapped, requiring the user to reenter the data. In Logo, the trick to doing this in elegant fashion is to think backwards!

Look at it this way: READLIST (or REQUEST) is an operation; it outputs a list of what the user types in. Here is a simple graphic of this action:

<-- READLIST

The arrow indicates the direction of the output, back toward the command or operation that called READLIST.

Let's take this Logo classic

```
TO BOX :SIZE
REPEAT 4 [FD :SIZE RT 90]
END
```

and use it to make an interactive box drawing program.

```
TO INTERBOX
PRINT [HOW BIG A BOX DO YOU WANT?]
BOX FIRST READLIST
END
```

The last line of the program executes like this. BOX waits for a number from FIRST, which waits for a number inside a list from READLIST, which waits for a number from the user. As long as the user types in a number, all is well. The number gets passed back to BOX eventually, like this.

BOX <-- FIRST <-- READLIST

However, if the user types in something other than a number, it gets passed into BOX where, within BOX, FORWARD protests that :SIZE isn't carrying a number.

Q and A continued

LOGO RECOGNIZERS

Logo has some built in "recognizers" that can inspect data objects and identify them. One of these is NUMBERP (or NUMBER?). It returns TRUE if its input is a number and FALSE if it receives anything else. We can use this primitive as a tool to trap erroneous input to INTERBOX when it runs READLIST if we arrange for a checking operation containing NUMBERP to inspect the data typed in before it can get inside the BOX procedure.

Remember that the data travels backwards from READLIST to FIRST to BOX. Suppose we stick in a checking procedure called CHECK.NUM in this chain of operations.

```
BOX <-- CHECK.NUM <-- FIRST <-- READLIST
```

CHECK.NUM intercepts FIRST's output before it can get into BOX, thus trapping bad data and allowing the good stuff to go through.

CHECK.NUM is going to take one input (a number, we hope!) and test it with NUMBERP. If the input passes the test, it is simply output to whatever calls CHECK.NUM (in this case, BOX).

```
TO CHECK.NUM :OBJ
IF NUMBERP :OBJ [OUTPUT :OBJ]
...
```

So far, so good. What should CHECK.NUM do if the input is not a number? Well, the user should be asked to type in good data.

```
PRINT [PLEASE ENTER A NUMBER.]
```

Now we want the user to type in a number, so we need another READLIST in here somewhere. But suppose that our user once again makes a mistake and types in a letter instead of a number? Oh, dear! We need to check that input, too!

Feels recursive, yes? Through the clever use of recursion, we can complete CHECK.NUM in one more line.

```
OUTPUT CHECK.NUM FIRST READLIST
```

Who! CHECK.NUM uses another copy of itself to check its own input! Makes one dizzy, this recursive stuff...

Bombproof programs can be written in Logo. By combining Logo's other recognizer operations (such as LISTP or LIST?) with the Boolean operations (AND and OR, or ALLOF and ANYOF), it is possible to filter any kind of user input.

Here's a challenge for you. Write a filter procedure called CHECK.RANGE which accepts three inputs: a number to be checked, a low number, and a high number. CHECK.RANGE should allow only numbers between the high and low value inclusive to pass through. If other input is received, it prompts the user to reenter the data.

(Ed. Note: If you have a question about Logo programming techniques or the thinking which leads to Logo programming, send it to NLX Q&A, PO Box 5341, Charlottesville, VA 22905. Each month, Jim McCauley answers selected questions in this column.)

Jim McCauley is a Coordinator of Computer Education for the Santa Clara (CA) County Office of Education, and has written Logo articles for many national publications.

NLXual Challenges

by
Robs Muir

What? If?

One of the oft repeated advantages of Logo over other educational computing languages is its extensibility. In this context, extensibility refers to the ability to expand Logo's "vocabulary" beyond its basic vocabulary by defining procedures. The name of each new procedure is a word which extends Logo's expressibility. Contrast Logo with what John Allen calls "general purpose" languages, such as BASIC.

Logo's original foundation rests on a core of primitives (or basic vocabulary) that forms the environment in which users write their procedures. As the language has matured, we are beginning to see versions of Logo with expanded sets of primitives to lower Logo's threshold, or ease of use. Most versions of Logo now include "primitives" such as ITEM, COUNT, and DOT, which could be written as procedures reasonably easily. Indeed, most Logo tutorials recommend learning how to write versions of these primitives as a good way to learn how the language works.

NLXual Challenges continued

WHAT'S THE QUESTION?

The question of importance here is, "Are there some primitives that you just can't do without?" What about IF? Can you write an IF procedure in Logo, or is IF a "primal" primitive?

Michael Tempel, as Director of Training for Logo Computer Systems Incorporated (LCSI), often presents the challenge of writing Logo primitives in Logo in the advanced workshops he teaches. After kicking around the IF question at the LCSI offices in new York and Montreal for several months, he and others have composed a number of ways of writing an IF procedure.

In the April issue of the NLX, Michael Tempel and Mario Bourgoin of LCSI will share the first few IF's which they and other LCSI staffers created. In the May issue, they plan to present the rest, along with a brief discussion of how Logo makes decisions. However, they would also like to see what our readers can do. Let's not disappoint them!

THE IF RULES

Here are some rules and guidelines in writing your IF procedure.

-No fair using TEST, IFTRUE (IFT), or IFFALSE (IFF), since they make up another form of IF.

-Write a two-input version (if-then) for starters.

TO IF.THEN :CONDITION :TRUE.LIST

-Then tackle the three-input version (if-then-else).

TO IF.THEN.ELSE :CONDITION :TRUE.LIST :FALSE.LIST

-This challenge doesn't officially contain an invitation for an IF which takes an optional third input. This would be quite difficult, since user-defined procedures require a definite number of inputs. You might think about the IF THEN ELSE form of MIT Logo (an older ALGOL patterned syntax), but that also would probably be very hard to write. Why do you think these are called NLXual Challenges?!

IF YOU.HAVE.A.SOLUTION [MAIL :SOLUTION] [KEEP.TRYING!]

TO MAIL :SOLUTION
SEND.TO
MICHAEL.TEMPEL
LCSI
555.WEST.57TH.STREET
SUITE.1236
NEW.YORK.NY.10019
END

TO KEEP.TRYING!
CO
END

Robs Muir is a teacher in the Claremont CA Unified School District and Claremont Graduate School.

Special Education Logo Users Group Announced

Barbara Luetke-Stahlman announces the formation of a Special Educators Logo Users Group. The objectives include setting up networks, sharing research results, and formulating grant proposals for Logo projects in special education. For more information, write to her at the Hearing Impairment Program, Northern Illinois University, DeKalb, IL 60115.

The National Logo Exchange, copyright (c) 1985 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 USA, 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 other correspondence to: The National Logo Exchange, Box 5341, Charlottesville, VA 22905. ISSN 0734-1717

Editor.....Tom Lough

NLXTRA

Attention overseas Logo users! Please send information on Logo books, articles, reports, and teaching aids available in your country for inclusion in the bibliography for the Logo 85 conference. Include publisher addresses and prices, where possible. Thank you. Bibliography, PO Box 5341, Charlottesville, VA 22905.

Do you present Logo workshops for teachers? If so, please let us know so we can publicize your activities in the NLX. We also support such presentations with Logo handout materials, upon request. Here is a workshop tip. At a recent Logo workshop, Glen Bull and Paula Cochran distributed to their participants copies of a Logo file disk which could be used by either an IBM PC or an Apple II computer. How did they do it? They initialized one side with the IBM DOS, then punched a hole in the edge and initialized the other side with Apple DOS 3.3! This technique could be used for any combination of two computers whose disk drives can read single-sided disks.

If you are teaching a summer course in Logo, let the NLX help publicize it. Send your course information to NLX, Attn: Summer Courses, PO Box 5341, Charlottesville, VA 22905.

The second edition of The National Logo Exchange's Logo Research Directory is now available. Editor Regina Sapona has done an outstanding job of organizing the more than 60 entries into operational categories, with cross-referencing. Each entry contains a research summary and descriptors, as well as a mailing address for more information. To receive your copy, send \$3 to NLX Research Directory, PO Box 5341, Charlottesville, VA 22905. If you are conducting Logo related research and are not listed in the directory, please enclose information about your work and interests.

Terrapin Logo is offered at a price of \$65 by Conroy-LaPointe, 12060 SW Garden Place, Portland, OR 97223. (800) 547-1289.

The NLX ABC's have been distributed to thousands of Logo teachers over the past two years. Now let's see what has been done with them! We would like to publish the NLX Book of Logo ABC's. If you or your students have used the NLX ABC procedures for Logo projects, send us a printout or a listing. We are especially interested in projects done with single letters, so they can be grouped in the A section, the B section, and so forth. However, there is also a section for multiple letter projects, too! Maybe you can combine some ideas from Cathy Frank's outstanding "wordle" article in last month's NLX.

The author of each project selected will receive a free copy of the finished book, and will be mentioned by name in the publication, if desired.

Michael Friendly has agreed to be the editor of this project. Send your ABC projects to Michael Friendly, Psychology Department, York University, Downsview, Ontario, Canada, M3J 1P3.

If you have not yet received your listing of the NLX ABC procedures and would like to be involved with this project, send a self addressed stamped envelope to NLX ABC's, PO Box 5341, Charlottesville, VA 22905. Be sure to specify your computer model and Logo version.

Logo Hawaiian this summer! Join Glen Bull, Paula Cochran, and Tom Lough for "Logo in the Classroom," a 3-hour graduate Logo course from the University of Virginia, to be taught at the famed Punahou School in Honolulu, Hawaii, July 8 - 19, 1985. The course concentrates on Logo applications in language arts, science, and mathematics content areas for teachers with Logo experience. For Logo beginners, an outstanding graduate introductory course is offered, taught by Elaine Blitman and Barbara Jamile. Make your plans early for these popular offerings. For more information, call (800) 223-2544. New York residents call (212) 686-5810 collect. Educational Spectrum, Inc., 21 East 40th Street, New York, NY 10016.

Comodore Logo is available for \$59 from Sunburst Communications, Room BN, 39 Washington Avenue, Pleasantville, NY 10570. (800) 431-1934.

"Integrating Logo Into the Curriculum," a mini-conference sponsored by ECOO-SigLogo, is scheduled for March 23, 1985, in North York, Ontario. Included in the program are eleven presentations on Logo in various content areas, ranging from special education to science to storywriting. For more information, contact Paul Davidson, Sloane Avenue P. S., 110 Sloane Avenue, North York, Ontario M4A 2B1 Canada.

The fifth annual National Microcomputers in Education Conference is scheduled for March 13-15, 1985, at Arizona State University. Many of the more than 200 scheduled sessions focus on Logo. For more information, write Gary Bitter, College of Education, Payne 216, Arizona State University, Tempe, AZ 85287, or call (602) 965-7363.

The Alberta Teacher's Association Computer Council announces ATACC 85, its third annual conference, to be held in Edmonton, Alberta, March 14-16. A number of Logo presentations are scheduled, including one entitled "Physics and Math Simulations in Logo," by Tom Lough. For more conference information, write to Anne McIntyre, 5640 Ada Boulevard, Edmonton, Alberta T5W 4N8 Canada.

EduComp announces a series of summer Logo activities. If you want to learn how to explore Logo with your students, and to obtain practical ideas for scheduling, using teaching aids, creating bulletin boards, and establishing a Logo environment, read on. Beginning Logo is taught June 26-28, starts from scratch and progresses through variables. Continuing Logo, June 28-30, contains more variables, recursion, procedure refinement, conditionals, and beginning list processing. Both courses are 30 hours long, carry two credits, and cost \$90 plus a \$15 lab fee. The instructors are Christi Jaeger and Carolyn Green, authors of Teacher, Kids, and Logo. For more information, write: EduComp Summer Logo, 14242 Wyeth Avenue, Irvine, CA 92714.

J B Software has developed two interactive Logo programs for use with handicapped children. Logo Learning with Numbers and Logo Learning with Words are available for Logo versions on the Apple and Commodore 64 computers. For more information, write J B Software, 5813 Castano Drive, San Jose, CA 95129, or call (408) 996-9630.

The January issue of The Jeffries Report, an informative newsletter which reports on the state of health of various aspects of the microcomputer industry, indicated that the new Commodore 128 computer has an empty read-only memory (ROM) slot inside. Various rumors are circulating for how the slot may be used. Jeffries suggests that the chip eventually installed in the slot might turn out to be one for Logo! This means that, when the computer is turned on, Logo is ready to go. Let's wait and see on this one! (The Jeffries Report, Box 6838, Santa Barbara, CA 93160)

The January 28 issue of InfoWorld had a story on page 15 about the new generation of computers Atari just unveiled. Featuring a 68000 microprocessor, the ST line has been called a lower resolution color Macintosh at half the price. The article suggested that the machines will be sold with either BASIC or Logo! This may be a computer family that bears watching.

In the January 1985 issue of School Microcomputing Bulletin, an interesting lesson plan is presented for teaching students about artificial intelligence. Logo teachers who want to emphasize this aspect of the language might find the lesson plan useful. For more information, write Learning Publications, Inc., 303 Bay Drive North, Bradenton Beach, FL 33510.

Dan and Molly Watt write that Learning with Logo (Byte/McGraw-Hill) is now being translated into Spanish, German, Japanese, and Hebrew, and that their Teaching with Logo (Addison-Wesley) is scheduled for publication this spring. Also, Dan's Learning with Apple Logo is now available.

The Project of Equal Education Rights (PEER) announces the establishment of the National Center for Computer Equity to help ensure that computer opportunities are available to all students regardless of race, sex, or disability. For more information, call Leslie Wolfe at (202) 332-7337, or write to PEER, 1413 K Street NW, Washington, DC 20005.

EXSYS announces Turtlelab: Music, a pack of cards containing instructions and musical activities for Terrapin or Krell Logo. The eleven-card pack costs \$7.95 plus \$1.00 for shipping. EXSYS, 2728 23rd Street, Greeley, CO 80631. (303) 330-8021.