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

LOGO EXCHANGE

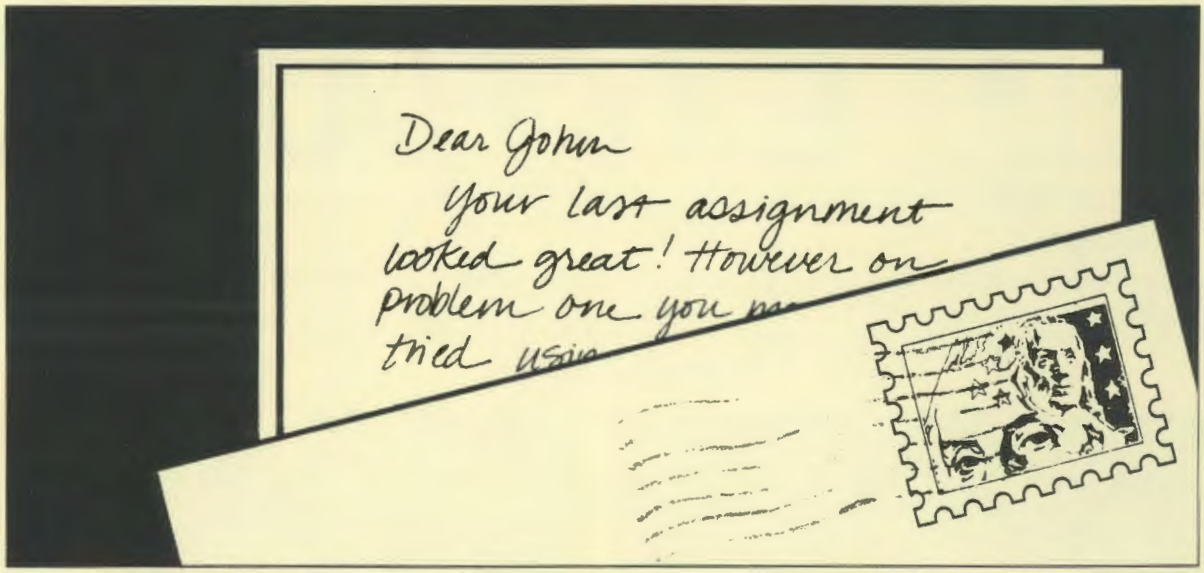
January 1990 Volume 8 Number 5

CONNECTIONS

International Society for Technology in Education

ISTE Publications

Finally, a long distance relationship that won't break your heart.



ISTE offers eight *Independent Study* courses that get to the heart of learning.

Each course thoroughly covers the title material and is designed to provide staff development and leadership training. You correspond directly with the course's instructor by mail, and can receive graduate credit through the Oregon State System of Higher Education.

Classes offered this year are:

- Introduction to Logo for Educators (available for LogoWriter or Logo PLUS)
- Fundamentals of Computers in Education
- Long Range Planning for Computers in Schools
- Computers in Mathematics Education
- Computers and Problem Solving
- Introduction to AppleWorks for Educators
- Computers in Composition
- Effective Inservice for Instructional Use of Computers in Education

Register for classes independently or with a group. Districts enrolling six or more teachers receive a fee reduction for each person enrolled.

Courses range from $222 to $336 for 3-4 quarter-hours of graduate credit. You have one year to complete your course.

Start a great long distance relationship today with an ISTE *Independent Study* Course.

Request an *Independent Study* course brochure. Write or call:

ISTE, Unviersity of Oregon,
1787 Agate St., Eugene, OR 97403-9905
ph. 503/686-4414

LOGO EXCHANGE

Volume 8 Number 5 Journal of the ISTE Special Interest Group for Logo-Using Educators January 1990

Publisher
International Society for Technology in Education

Advertising space in each issue of *Logo Exchange* is limited. Please contact the Advertising Mgr. for availability and details.

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

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

Contents

ISTE Membership

	U.S.	Non-U.S.
	28.50	36.00

SIGLogo Membership (includes *The Logo Exchange*)

	U.S.	Non-U.S.
ISTE Member Price	25.00	30.00
Non-ISTE Member Price	30.00	35.00

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

From the Editor

Connections...and then what?

The decision to have several "theme" issues during this publication year seemed an exciting idea last spring when I first made the announcement. I have since discovered that a commitment to theme issues can be stressful to the editor. What if, I worry, no articles on the theme are submitted? What if articles that were promised don't arrive by the deadline? And what will the columnists produce? Will they be able to maintain the tone of their column while still connecting to the theme of the issue as a whole.

If you received the October "Just for Beginners" issue, then you know that most of the material in that issue was indeed appropriate for beginners. That issue certainly gave me confidence that it was possible to produce an issue devoted to a theme without acquiring too many more gray hairs. However, my anxiety about this current issue increased dramatically when I returned from a professional trip only to find that not only were several of the columns late, but two of the promised articles had not arrived. Within a couple of days I was breathing again as one of those articles arrived via express mail and the other via email from half way around the world. At the last moment the issue fell beautifully into place – with more material than the journal could hold!

I think you will find that there is a richness in the collection of material in this issue that would be hard to match anywhere: from classroom connections through technical information; from research to teacher training. But more than the specific projects and ideas discussed in these pages are the issues raised. Thinking about those issues caused me to reflect on the changes in the role of Logo in computer education over the past decade.

Some 8 or 9 years ago, I first introduced Logo to my school system via a special project in one of our elementary schools. We had only a few computers scattered through our system and this project was an opportunity to try something new. It wasn't long until Logo spread throughout our grade schools and into our secondary schools. Even though we continually added equipment and copies of Logo, our resources were limited. One computer to a classroom was an unusual luxury; more often 2 or 3 classrooms shared a single computer.

In the early 1980's, most computer equipment was quite expensive, dramatically limiting the number of computers a school could afford. However, the software was also very limited. The choices were often between poor quality CAI and BASIC programming. During those years, I recommended to my school system, as well as to many others with whom I consulted, that it was appropriate to use *only* Logo in their computer education program. Logo provided an open ended environment that would work with any age level, could be used with a limited amount of equipment, and was relatively inexpensive in a day when few school systems provided money for software. (If only we had had LogoWriter or Logo PLUS then, my advice would have been even more appropriate.)

But times have changed. The software available for school computers is rich and varied. Generic tools such as word processors and databases are widely available for all age levels. Problem solving software abounds. There are simulations that allow exploration in a variety of subject areas. Newer products such as HyperCard provide Logo-like environments. Further, as you will see in this issue of *LX* , Logo itself can be linked with all manner of other technologies and Logo text and graphics can easily be exchanged with text and graphics from other pieces of software. Logo no longer stands in isolation as one of the very few pieces of excellent software as it did 10 years ago.

So what of the next decade for Logo? Logo "purists" would have us continue to insist that Logo in and of itself will revolutionize education and that there is no need for other pieces of software. Increasingly, I believe, those purists are standing alone. Those of us in the Logo community need to be open to new Logo-like environments – places that provide the same kinds of exciting learning experiences that Logo has provided in the past. Does that mean we abandon Logo? Certainly not. Logo is still the best tool for many learning activities, as you will clearly see in the pages of this issue.

So, as you read through the pages of this *LX* think about what you believe the role of Logo and *LX* should be in the future. Read with care the last part of Glen and Gina Bull's "Logo & Company" column and cast your vote for or against their concept of *Learner-Based Tools*. Spend some time thinking about where Logo fits in the broader scheme of things – and then share your thoughts. Glen and Gina and I would all love to hear your opinions.

Sharon Yoder
SIGLogo
ISTE
1787 Agate Street
Eugene, OR 97403
CIS: 73007,1645 BITNET: Yoder@Oregon

Monthly Musing

Connections: The Process of Procedures

by Tom Lough

I have learned to enjoy and appreciate telecommunications and electronic mail. For example, I send this column to Sharon by email each month. When I call up the mainframe computer, it signals that it is ready for operation by sending a single word to my computer screen:

```
CONNECT
```

Seeing that word appear always excites me, because I realize that I will be in electronic touch with other Logo colleagues in the next few moments.

When I am writing this monthly column, I always have a similar sense of excitement, because I realize that I will be in touch with thousands of other Logo colleagues when the words I write become part of an issue of *Logo Exchange*. That excitement of connecting is one of the major reasons I started *LX* in the first place. So you can imagine what it meant to me for the theme of this issue to be "Logo Connections."

I began thinking about connections within Logo. One of the first things that occurred to me was how procedures are connected to each other in a superprocedure. If we have a superprocedure such as

```
TO HOUSE
SQUARE
TRIANGLE
END
```

it is clear that SQUARE and TRIANGLE are connected by virtue of the sequence in which they are called. In particular, the end of the SQUARE procedure is connected to the beginning of the TRIANGLE procedure by the calling sequence of the HOUSE superprocedure.

Sometimes it is necessary to connect procedures a little differently within a superprocedure, such as the following.

```
TO HOUSE
SQUARE
FORWARD 50
RIGHT 30
TRIANGLE
END
```

In this case, SQUARE is connected to TRIANGLE through the intermediate commands of FORWARD 50 and RIGHT 30. Commands such as these prepare the way for the next procedure to carry out its instructions.

Many teachers feel that they are connected to their students like this.

```
TO EDUCATE
TEACHER
STUDENT
END
```

The TEACHER procedure is connected directly to the STUDENT procedure. First the teacher does something, followed immediately by the student.

Others may suggest that such a procedure should be written as follows.

```
TO EDUCATE
TEACHER
FORWARD 50
RIGHT 30
STUDENT
END
```

In this case, the action of TEACHER is connected to STUDENT by the intermediate commands of FORWARD 50 and RIGHT 30. Commands such as these prepare the way for the next procedure to carry out its instructions.

The Logo HOUSE example was probably clear, but this one may be somewhat muddied. Could FORWARD 50 and RIGHT 30 represent any aspects of the education process which connects the teacher and the student?

Perhaps these commands might represent an adjustment in the mind of the student. Or, maybe they could indicate the passage of some time. Perhaps they stand for the effect of intangible qualities, such as patience or trust or caring.

It really doesn't matter if we can't figure out what they mean. The important thing is that we should think about the connections between teachers and students as something special. And if musing with a Logo procedure helps, then so much the better. **FD 100!**

Tom Lough, Founding Editor
PO Box 394
Simsbury, CT 06070

Logo Ideas

There are connections...and then there are connections.

by Eadie Adamson

connection: a junction; a joining; a shared space
Preliminary definition (EA)

How do you define the word **connection**? Logo connections come in many forms. For instance, Logo can be used to control many external devices. They are fun to explore and play with. Of all the possible connections of this type, the one most fascinating for my students has been the Lego-Logo connection.

My students were all familiar with LogoWriter and had used LogoWriter to write their own motion games. When they began working with Lego TC logo in the classroom, it was natural for many of them to want to exercise control over these cars or conveyor belts or turtles, in the same way in which they had controlled the LogoWriter screen turtles in their game programs.

It didn't take long before one student began working to adapt the Lego procedures for moving the Lego floor turtle. Ted wanted to use the arrow keys rather than the commands provided on the Lego "Turtle" page. He wanted the program to work just the way one of his motion games had worked. He named his own page of turtle stuff "Teds.Turtle" and, since he knew what he needed to do and saw the connection with his previous experience, went to work very independently. His initial program used the arrow keys to move the turtle forward, back, left and right. To do this he needed to know the ASCII value of the arrow keys . He checked this by typing in the Command Center:

```
show ascii readchar
```

He then pressed Return, followed by the arrow key he wanted to program first. The ascii value of the key in question appeared in the Command Center. (**Readchar** is a primitive which reads a character pressed at the keyboard.) Ted could then access the key by addressing, for example, the up-arrow key as **char 11**. His initial program was constructed fairly quickly. Each arrow key was to send the Lego floor turtle in motion, either moving forward, back, left, or right. Ted planned to use the ready-made procedures (**tfd**, **tbk**, **tlt** and **trt**) provided on the TURTLE page on the Lego disk to help him with this part of the program. Ted's program looked like this:

```
to turtle
name readchar "key
if :key = char 11 [tfd 10]
if :key = char 10 [tbk 10]
if :key = char 8 [tlt 2]
if :key = char 21 [trt 2]
turtle
end
```

Next Ted's attention turned to the touch sensors on the front of the turtle. He incorporated them into his "turtle" program:

```
to drive2
listento 6
if sensor? [tbk 15 tlt 5]
listento 7
if sensor? [tbk 15 trt 5]
end
```

He inserted **drive2** into his turtle program so that it read like this:

```
to turtle
name readchar "key
if :key =  char 11 [tfd 10]
if :key = char 10 [tbk 10]
if :key = char 8 [tlt 2]
if :key = char 21 [trt 2]
drive2
turtle
end
```

Now the turtle was smart enough to back away from an obstacle, then turn a bit before proceeding.

This was only the beginning for Ted, though. He proceeded to redesign the turtle slightly (it had been built following the instructions which come with the Lego TC logo kit) by adding lights which turned on when the turtle was used. Then Ted decided the turtle's lights should operate as warning lights: some should go on when the turtle moved forward, others when the turtle moved backward, and the appropriate lights should be lit when the turtle turned.

Further Connections

Ted's variation on the classic "instant" program gave me some other ideas, which brings up a further Logo connection: creating and programming for an audience. While completing

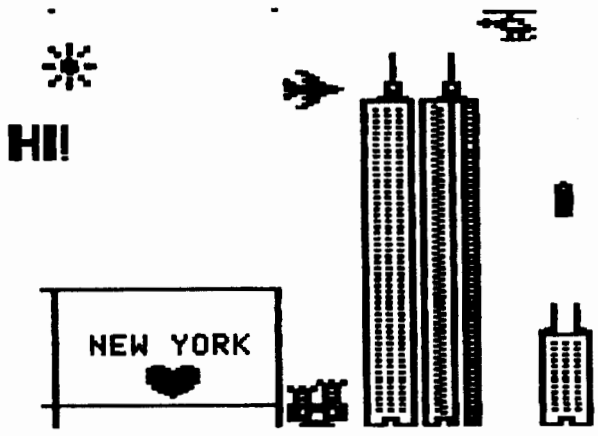
a project such as Ted managed to do can be satisfying, Ted ultimately got more satisfaction from showing it to his classmates, especially those who had not yet used LEGO TC logo. This year one of my classes will build the floor turtle, make an "instant" program for it, and then have a chance to teach our kindergarten students how to use it. This prospect of a further audience will give my students an additional goal towards which to work. We might also plan a "Lego Show" and invite their parents to attend, something that happened very informally last year.

Although Ted had not begun his project with this in mind, having an audience was an important component of his work. Increasingly I think that the further extension of creating a project with an audience in mind is a connection we often fail to make, although it can be the best Logo "connection" of all. Part of the idea of working with Logo has been to build a sharing community. Too often this may happen in a particular classroom, but the sharing stops there. Why don't we, instead, spend just a little more effort and take the small amount of time to share our Logo work with other classes? Our school encourages, through a "Big Brother" program, a kind of sharing with older children helping occasionally in younger children's classes. As I observed these interactions last year, I began to think that this was the best of all connections, children helping and sharing with other children.

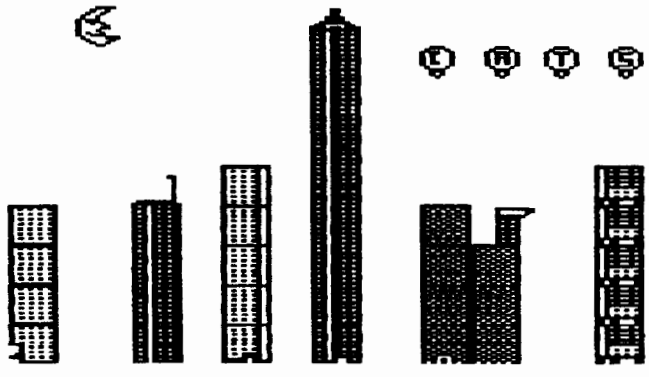
A Different Kind of Classroom Connection

Last year I noticed during one of my regular visits to "play" with LogoWriter with the kindergarten class that the boys were beginning to learn to read and write simple words. At about that time my fourth grade classes were creating projects of city scenes with flashing signs, thus "connecting" a LogoWriter programming idea with signs they see in the real world. Aha! a Logo connection idea was formed: the fourth grade boys could design special sign pages which could be activated by some of the words the kindergarten boys were learning.

Originally, when the fourth grade and I talked about the idea, we all thought we would simply create the programs and then lend them to the kindergarten to use. This, however, would have deprived the older boys of the pleasure of truly sharing their work. I suggested to them that when their projects were ready we might arrange for the kindergarten to visit our class so that the sharing could take place individually. What an incentive this was! Creative ideas sprung up everywhere (See the illustrations with this column.). We needed to plan how we were going to present each program or page of commands. In addition to polishing up their project – because they now had a real audience in mind – the boys became truly concerned about making it easy for the younger children to understand what to do. They decided to make cards to sit at the top of each keyboard on which they would print in big letters the words to type to get a sign to flash or, in some cases, to start an animation.

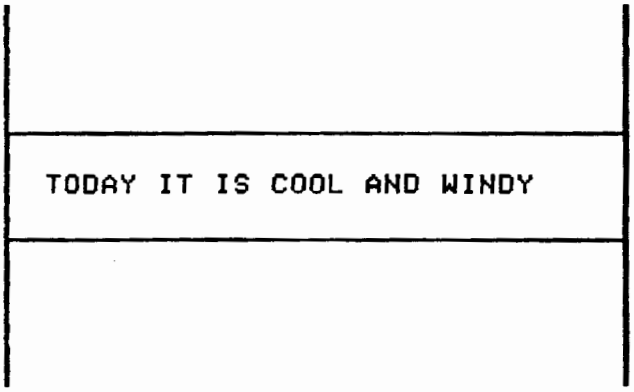


Some of the boys thought the visiting day would *never* come! Finally we arranged a morning visit, half of the kindergarten class at a time. That gave each of the older boys one or two younger boys to work with. After a reasonable period of time, we played "musical chairs" of sorts: each of the younger boys moved to the computer on the left and began to work with a different older boy.



All too soon the class time for the younger boys ended. The older boys still had a little time left. I decided that a debriefing would be most helpful. I asked them to think about the experience and to think about what surprised or disappointed them, what happened that they didn't expect, and how they felt about it. The boys made some wonderful observa-

tions: the younger boys liked things they had thought were "too dumb" to be bothered with such as simply using the turtle-move keys to move an object about on the screen; they were able to do things such as creating shapes – that the boys had thought would be too hard; they found that they needed to figure out what to do with the boy who "wouldn't behave" (!); what they expected the kindergarten boys to enjoy the most was often what they were least interested in doing. This kind of closure on the experience gave us a chance to reflect together on the problems of being a teacher in a Logo environment.



This Logo connection was so successful that these same students have asked if they can do something similar this year! Not surprisingly, the kindergarten teacher has also asked if we can plan more sharing. This has made me think more about sharing with other classes as well. Interactive or rebus stories could be created and shared with first and second grade students, for instance. Older students could "write" stories told to them by younger children and then use LogoWriter to illustrate, animate and/or automate the tales.

Question: Do all Logo connections have to be hardware connections? Not for us!! What about you? How do *you* define connection?

Eadie is a computer coordinator at The Allen-Stevenson School, an independent school for boys in New York City.

Eadie Adamson
1199 Park Avenue, Apt. 3A
New York, N.Y. 10128

Lego-Logo For Elementary

Introducing Lego-Logo to Elementary Students

by Kwok-Wing Lai

The recent advent of inexpensive robots to the elementary classroom has great educational potential. It has been suggested by some educators (e.g. Bitter & Gore, 1986; Neighbours & Slaton, 1987; Pearce, 1987) that computer controlled machines enable learners to carry out investigations involving both heuristic (a rule of thumb that often, but not always leads to a solution) and algorithmic (procedures guaranteed to result in the solution of a problem) problem solving, and apply them to situations involving feedback control. Particularly for young learners, robot oriented activities provide a concrete, and a learner-centered environment conducive to concept development.

Recently some programmable systems are available in the market where users can assemble their own models and machines and interface them with a computer. For example, the Lego Technic Control Set (Lego Australia Pty. Ltd., 1987) allows users to build fairly complex models by using regular Lego elements and special elements such as motors, optosensors, counting discs, light bricks, electric cables, steering elements, and wheels available in the set. These models could then be interfaced with a computer by using an interface box (or card) and operated by a programming language such as Logo. Since many elementary students are already familiar with Logo and Lego, a learning environment could easily be set up to help them acquire a deeper understanding of machines and control technology, and improve their skills in planning and co-operation, manipulation, measurement, logical thinking, and evaluation.

Lego-Logo and Problem Solving

Problem solving skills and strategies are best learned in a concrete situation. The problem learners tackle in a Lego-Logo learning environment such as building a sliding door is concrete and relevant to their life experience. Therefore problem solving strategies could be easily introduced and practised. In my own study (Lai, in press) I worked with a group of New Zealand elementary students (aged 10 to 11) and introduced problem solving skills and strategies to these learners as we were building Lego models. In the eight sessions (one and a half hours each) they were encouraged to practise Polya's (1957) problem solving procedures of understanding the problem; planning; implementing; and evaluating solutions.

Lego-Logo and Metacognitive Awareness

Current research in learning suggests that effective learning depends on having knowledge of the repertoire of one's cognitive abilities and being able to regulate one's cognition, e.g. planning, checking, and evaluating learning strategies and outcomes. Brown, Campione, & Day (1981) contend that learners need to be aware of the importance of applying problem solving strategies to enhance understanding. The Lego-Logo learning environment provides an opportunity for the teacher to introduce metacognitive awareness to the learners, and also encourage them to exercise self-reflection by asking themselves metacognitive-sensitive questions such as "Am I on the right track" or "Do I need more information to solve this problem".

Lego-Logo and Cooperative Learning

A Lego-Logo environment also encourages cooperation. Not only the hands-on experimentation encourages a lot of sharing and discussion among group members, different roles learners need to play in this environment also enhance their appreciation of the need to work together in order to solve their problem. In my study four major roles (model builder, programmer, questioner, record keeper) were assigned to the members in each session.

How to start?

Some basic concepts of control technology need to be explained at the outset. Students need to understand that a computer could easily communicate with an external device such as a printer or a motor, by using a port (usually) located at the back of the computer. The best way to start is to give a demonstration by connecting the computer to the interface box (basically a switch system where a single bit of data coming out from the computer turns a switch on or off) and then hook a small device such as a light bulb or a motor to the interface box. The Lego interface box has six output bits and two input bits where you could output data from, and input data to the computer. A simple Logo procedure could be written to operate this device. Commercial program such as Control Logo (Logo augmented with a library of special control procedures) could make life even easier. Students only need to learn a few commands. The following is a procedure I used in the first lesson of my study, written in Control Logo:

```
To Demo
Turnon 0        (a motor connected to Bit 0 is turned on)
Wait 40         (the motor keeps going for 40 units of time)
Turnon 3        (a light bulb connected to Bit 3 is turned on)
Wait 40         (keeps going for 40 units of time)
Turnoff [0 3]   (both the motor and light bulb areturned off)
Turnon [2 4]    (another motor and light bulb are turned on)
Wait 40
Turnoff [2 4]
End
```

If the students are familiar with Logo they should have no problem in understanding this procedure. In my own study about half of the students had no Logo experience and they found little difficulty learning to write Logo programs. Students will be very excited at this point but time should be allowed for adequate discussion on the use of problem solving strategies in learning. Also, they should be reminded to monitor their learning process.

To further familiarize the students with the use of special Lego elements, it is helpful for them to build the first one or two models under the guidance of the teacher. In my own study students were asked to build a traffic light system to control a cross road traffic, and to build a washing machine by following instructions provided by the Technic Control Set. An example of the procedure of the traffic light system is listed below. In this model two sets of traffic lights are placed perpendicular to one other, directing traffic at an intersection. In Set A the green light bulb is connected to Bit 0, amber to 1, and red to 2. In Set B the green light is connect to Bit 3, amber to 4, and red to 5.

```
To Traffic
Turnon [0 5]  (The green light on Set A and the red Set
Wait 200                light on B are turned on)
Turnoff 0
Turnon 1      (The amber light on Set A is now on)
Wait 80
Turnoff [1 5] (The red light on Set B and the amber light
Turnon [2 3]            on Set A are now off; the red light on
Wait 200                A and the green on B are on)
Turnoff 3     (The green on B is off and the amber is then on)
Turnon 4
Wait 80
Turnoff [2 4] (The red light on A and the amber on B are off)
Traffic
End
```

To construct this traffic light system, the students first of all had to figure out how a traffic light system worked. At this point they were encouraged to represent their problem in different ways, e.g. using a diagram. When they have figured out the logic of the system they could start building the model by dividing the tasks into sub-tasks. Then finally they programmed the system to make it work. In this process the learners had ample opportunity to plan, to use different strategies to tackle their problem, to share their ideas, and to evaluate their work. My experience is that students are very

keen to experiment and build their own models. The role of the teacher is only to encourage them to use a problem solving approach to design and build their own models, but not too much intervention.

It is no doubt that a Lego-Logo learning environment is highly motivational to young learners. In my study both girls and boys were enthusiastic about the project and no gender difference was observed in groups. The environment was stimulating because the participants were able to control their learning process and the concreteness of the problem tasks enabled them to relate to what they had encountered before. It was also clear that this learning environment was conducive to acquisition of knowledge in control technology and skills in problem solving. It also promoted metacognitive awareness. The cooperative atmosphere encouraged the participants to acquire knowledge in a more active and constructive way, as well as sharing ideas and helping each other. The following comments may give you some ideas of what they reacted to this learning environment:

> I've learned that when you're working with Lego that it is possible to make anything. Because if you think hard you will be able to make anything work...
>
> I have learned how to work with Logo a lot better than I used to and learned how to solve problems quicker than I used to ...
>
> [I learned] how to work together as a team and to co-operate together and to decide in thing together.
>
> Working in a group is a lot easier because you've got other people in your group who may know more than you. It was pretty good when I got stuck because somebody always seemed to know the answer
>
> ...problem solving that will help me in future...to understand the problem...to see the problem and to [fix] it out

If there is so much fun in building a Lego model, and if so much could be learned, why don't we give our students a try?

References

Bitter, G. & Gore, G. (1986). Robots in the classroom: Another of tomorrow's teaching tools today. *Computers in the School*, 2(4), 15-20.

Brown, A., Campione, J., & Day, J. (1981). Learning to learn: On training students to learn from texts. *Educational Researcher*, 10 (1), 14-21.

Lai, K. (in press). Using robots in primary classrooms. *Proceedings of the Third New Zealand Computers in Education Conference.*

LEGO Australia Pty. Ltd. (1987). Something about LEGO Technic Computer Control Sets. *Com 3 Journal*, 13(3), 21-22.

Neighbours, D. & Slaton, F. (1987, March). Robots in my classroom? *The Computing Teacher*, 24-28, 45.

Pearce, J. (1987). Robotics with Lego and Logo. *Com 3 Journal*, 13(3), 18-20.

Polya, G. (1957). *How to solve it: A new aspect of mathematical method* (2nd Ed.). N.Y.: Doubleday.

Dr Kwok-Wing Lai teaches computer applications in education courses at the University of Otago, New Zealand. His current research interest is in problem solving. He can be reached at

Department of Education
University of Otago
P.O. Box 56
Dunedin, New Zealand

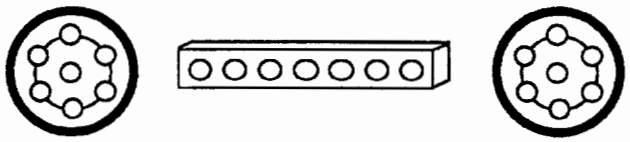
About the Cover

The pictures on this month's cover were taken during the Science and Whole Learning Workshop at the Media Lab at MIT. See page 15 for the article which describes the work done there.

Beginner's Corner

Back to the Future—The New Generation of Floor Turtles!

by Dorothy Fitch



Once upon a time...

When Logo was developed in the mid-1960's at MIT, there was no familiar turtle shape that lived on the computer screen. In fact, back then there were hardly any computer graphics monitors at all, at least not affordable ones! In those dark ages, computers often looked more like a combination computer/printer. What you typed at the keyboard was printed on paper, not on a computer screen.

So in the beginning, Logo commands were given to an object shaped like half a basketball—a robotic floor turtle. You could tell it to go forward or backward and turn it left or right. It had a pen to draw with and commands you could use to pick the pen up and put it down. The graphics world of Logo at that time was three-dimensional and larger than life. Kids could gain a real sense of movement and directionality. Their "buggy" commands had vivid and dramatic consequences. Group planning and cooperation was imperative. And talk about excitement in the classroom!

A turtle by any other name...

By the time Logo entered schools *en masse*, it had taken advantage of new technology – microcomputers and color graphics monitors that could show a turtle on the graphics screen. Wow, what progress! This certainly made things more convenient and less costly; there were fewer classroom management issues to deal with and fewer wires to trip over.

Progress in one sense, to be sure. But what are we missing now? The turtle on the screen is more distant from us and quite unapproachable. We can't walk around it and view it from different perspectives. We can't follow it or pace out distances to help us estimate. And we can't see how it moves and turns. The screen turtle zips from place to place and turns so quickly we can't tell whether it is going clockwise or counterclockwise; we can't even tell it if makes 16 complete revolutions before stopping! The screen turtle is intangible, made up of little bits of phosphor, not *real* things like gears, building blocks and motors.

Isn't it time to have the best of both worlds – a turtle on the screen and one on the floor? Original-style floor turtles are still available, but today there are also many more options for connecting robotic devices to Logo. These robots can assume a variety of shapes and they have more capability and features.

What will robotics do for my students?

Whether you are using robots to introduce Logo to younger children or to explore computer science and engineering with older students, these devices can open up a new exciting world for you and your class.

For younger children, robots provide immediate feedback in a way that makes sense to them. Some children don't readily or immediately identify with a two-dimensional object on a computer screen. They learn through imitation and variations of the familiar. By following the turtle as it moves around the floor, they gain a sense of direction (and confidence) that will help them when they use the Logo screen turtle.

Intermediate students learn how gears work, how to follow construction plans, how to write and refine a Logo procedure to perform a task, how to plan an original project. They invent machines, create new tasks, solve problems. They get excited about learning, they stick to their task until it is completed, they get involved. They become inventors, designers, experimenters and scientists.

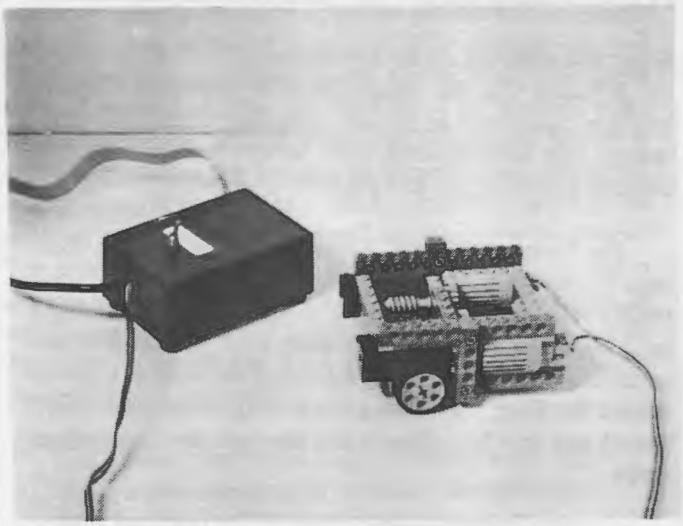
Robotic devices teach older students about precision, debugging, decision making, estimating, electronics and much more. Robots are an exciting introduction to the world of programming and engineering. Students begin to understand the important roles that robots can play, from performing rote jobs in industry to assisting the physically handicapped.

What does a robotics package include?

We're going to describe Terrapin Robotics, a product designed by Technology Services Institute of Connecticut with programs written in Logo by Terrapin. These kits are designed to be easy to use for beginners, yet complex enough for sophisticated programmers.

The kit comes with an interface box, which has its own low voltage DC power supply that plugs directly into an outlet. The interface box has an On/Off switch for controlling the motors, and a 16-pin connector that plugs into the internal

game I/O port of an Apple II+, IIe or IIGS. (A kit for the IBM will be available soon!) Also included is a Zero Insertion Force (ZIF) extender connecter, which conveniently brings a 16-pin plug outside the computer.



The kit works with a variety of motorized constructions sets, including Lego, FischerTechnik, Robotis, Erector and Capsela. With the package comes the Lego Universal Buggy Kit #1038, which the manual and software use for purposes of explanation and demonstration. There are two motors, a variety of gears and plenty of Lego pieces for building the car or other creations you invent. You are limited only by your imagination and the number of pieces you have to build with!

The 72-page Instruction Manual gets you started from the very beginning. You learn how to unpack the box, copy the program disk, build the car, connect it to the computer, and test it. The manual also provides instructions for controlling the car, calibrating it to run properly on your surface and writing and saving your first program. Software listings are included for the sample programs that are on the disk, as well as a variety of lesson ideas, and topics for the advanced user.

The double-sided disk contains sample programs (both DOS 3.3 and ProDOS) written in Logo and BASIC. These programs help you test and control your car as well as serve as models for programs you will want to write on your own.

A second kit is available that includes three light sensors. These sensors read light levels and return information through the game paddles 0-2. They can detect differences between colors, not just distinguish black from white. With light sensors, you can teach the car to follow a path, stop when it finds a dollar bill or other colored object, react when it senses a certain level or light, and so on.

How do I get started with the kit?

The first step is to build the car from the Lego kit that comes with the package. Of course, you can build other things, but a car (or buggy, as the Lego folks call it) is something that children can identify with readily and it makes a good starting project.

When I first tried out the package, it took me about an hour to follow the directions for building the car, hook the car to the computer and get it to pass its test drive successfully. Once you have done it a few times, it only takes about 20 minutes. Your students will have fun putting it together themselves, and it is an interesting project for them to follow the directions, which are from Lego and are in picture form. It is a challenge in spatial relationships to be able to tell from the pictures how the car should look from all angles at each stage of the construction! Unless your students are very young or you have an extremely limited time with which to work, your students should build the car themselves, rather than your doing it for them. (However, you may want to build the car yourself first and then take it apart, so you can help out if your students get into trouble!)

Once the car is built, you then attach it to the computer. The ribbon cable from the interface box plugs into the internal game I/O port, so you can use it on an Apple II+, IIe or IIgs, but not an Apple IIc, IIc+ or Laser computer. The Zero Insertion Force extender cable makes it easy to plug it and unplug it from the computer without having to remove the computer's cover.

The test program will let you know if everything is plugged in right, and the demo program, written in Logo, will send the car in a variety of cross, square and hexagon designs. Now that you have a car that understands commands, you can try your own instructions and experiments.

There are several Logo programs on the disk that help you move the car in different ways. First begin with the CAR.LOGO file, which contains the following commands. (Since these commands are actually procedures written in Logo, you can rename them, edit them, or use them in REPEAT statements or procedures you write.)

CARFD 4	moves the car forward 4 inches
CARRT 90	turns the car to the right 90°
CARBK 9	moves the car backwards 9 inches
CARLT 45	moves the car to the left 45°

A few activities to try!

- Place the car along a yardstick and practice addition and subtraction by sending the turtle to different inch marks. Use the Logo program called TIMED, which allows you to send the car forward or backward for a specific length of time. How many seconds does it take to travel 8 inches? How many seconds does it take to travel two feet?

- Create a mini-obstacle course of coffee cans and lunch boxes for the car to negotiate. Divide the class into teams and provide an equal amount of practice time for each. Then set up a race where the teams compete for the best time.

- Place cards numbered from 1 to 5 on the floor and program the car to pass over each card in order. First use only CARFD, CARRT and CARLT commands to move the car. Then try using only CARBK and one turning command (either CARRT or CARLT).

- Draw a floor plan of the school or a map of the neighborhood on a big piece of paper and place it on the floor. Practice map and direction skills as you pretend to deliver mail to various locations.

- If you have two kits, try to maneuver the two cars so that their front bumpers are as close as possible, but not touching. Or play "Follow the Leader – whatever car 1 does, car 2 must imitate!

- Create a garage for the car and see how few commands it takes you to "park" the car there.

For more adventures...

Here is a complete list of the sample Logo programs that come with the package:

DEMO.LOGO	a menu-driven program for testing and demonstrating the car's capabilities.
CARS.LOGO	procedures for moving and turning the car (CARFD, CARRT, etc.).
TIMED.LOGO	procedures for moving and turning the car for length of time (in seconds).
SPEED.LOGO	procedures for moving and turning the car at different speeds.
MOTORS.LOGO	procedures for controlling single motors so they can work independently.
FIGURE8.LOGO	an example of controlling individual mo tors to move the car in a figure-8.
PATH.LOGO	an example of how to program the car to follow a path using light sensors.
TEACH.LOGO	here the user determines what action the car should take based on light levels.

Of course, you are always encouraged to write your own Logo programs!

For the advanced user, the manual explains how use the .DEPOSIT commands to operate the motors, vary duty cycles to reduce the motor speed and yet maintain full torque, program turns using three different approaches, and connect the interface to other motorized kits.

So...

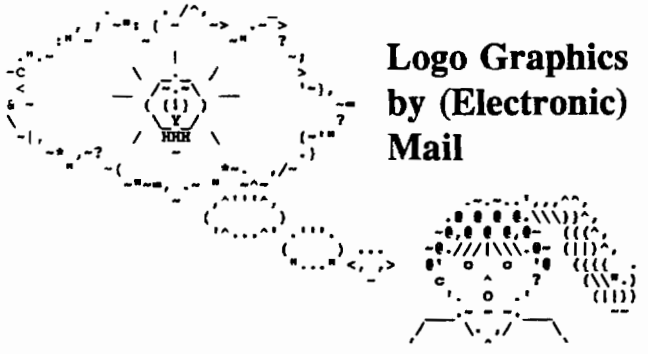
Robotic turtles aren't new. Why not start the new year by giving your students a taste of how Logo started and where it is going? Get the ultimate Logo shape editor—a kit that lets you construct and control your very own 3-D "turtle!"

For more information about Terrapin Robotics Kits, contact Terrapin at 400 Riverside Street, Portland, ME 04103, 207-878-8200.

A former education and computer consultant, Dorothy Fitch has been the Director of Product Development at Terrapin since 1987. She can be reached at:

Terrapin Software, Inc.
400 Riverside Street
Portland, ME 04103

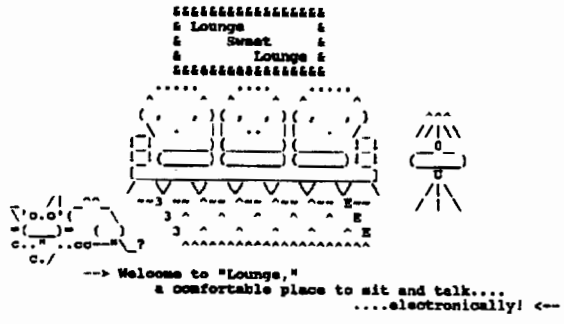
Logo LinX



Logo Graphics by (Electronic) Mail

by Judi Harris

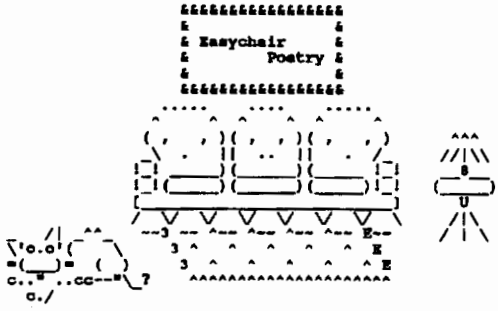
Have you ever seen a piece of "ASCII art"? Here is one that welcomes teachers to a public online computer conference in central Virginia called "The Lounge."



As you can see, this picture is made up of text characters. It was (painstakingly) created with a word processor. It is sometimes called "ASCII art" because the characters that comprise it are recognized by all types of computers; they are part of an across-manufacturer "American Standard Code of Information Interchange (ASCII)."

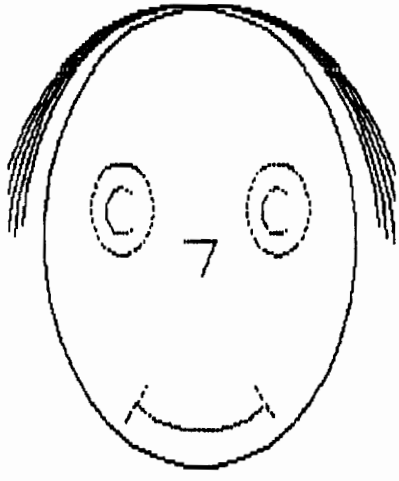
Electrified Art

If I sent this picture to you as an electronic mail message via the modem connected to my IBM PC, you could view and capture it with your Apple IIe, or any type of personal computer connected to a telephone line with a modem. You could save the Lounge welcome screen to your files diskette, load it into your Apple word processor, erase the welcome message, change the sign to read "Easychair Poetry," and then print it out on a ditto master. In this way, the revised Lounge greeting might serve as the title page for your class' most recent creative writing publication.



Pixelated Problems

Many electronic mail systems, online computer conferences, and electronic bulletin board systems will accept, store, and transmit only text characters such as these, or files written in ASCII form. In these electronic forums, only ASCII art can be directly exchanged. Logo graphics, such as this picture drawn by fifth-grade student Patrick Reynolds, are comprised of dots, or *pixels*, instead of text characters, and therefore cannot be transferred directly over many electronic networks.



If Patrick, who lives in Virginia, wanted to send his picture to an electronic penfriend in California, he could transmit the Logo procedures that he used to create it, but if the recipient uses a different version of Logo than Patrick does, translating may be difficult and time-consuming. It is possible, though, to convert a picture such as this one, pixel by pixel, to a text character code that can be transferred electronically, then decoded into pixelated form by whomever receives it. Developers at CompuServe, Inc., a commercial telecommunications service, have developed two picture exchange formats to do this.

RLE, or "Run Length Encoded" format, and GIF, or "Graphics Interchange Format," are two systems that can be used to encode and decode graphics images such as those created with Logo and saved with a SAVEPIC command. GIF pictures can be encoded and decoded in color; RLE pictures decode only to black-and-white images. Unfortunately, although GIF-decoded pictures can be displayed on Apple II series computers in color, Logo pictures created on Apple IIe's and IIc's can be encoded in RLE format only, and therefore decoded after electronic transmission in black-and-white only. It is possible, though, to encode and decode pictures for the IIgs in full-color GIF format.

Here is what Patrick's picture "looks" like after being translated into RLE code.

```
H
                GS
(+*6*, y,*&6&*~ m+*&B&*, d+)%N%*+
]*)$X$** V*)$`$)+O*($h$)*H*($p$)*
A*)"x"** <"!"!$)"|"*$!"!" 8"!"!$)"
!"*$!"!" 4"!"!$)" %"*$!"!" 0%!$)"
)"*$!% ,%!$)" -"*$!% (%!$)" l"*$!%
$%!$)" 5"*$!%%!"!!*!
9!+!!"!%{%"!!")" ;"*"!!"%w"!!!!!$*!
?!+!!"!!!!!"t!!&!!+!
A!+"!!!$!!q"!!!!!!!"*"
C"+$!!!!!"n!!"!#!!+!
G!,!!#!!"!k"!!!#!"*"
I"+"!%!!i!"!!#!!+!
M!,!!#!!!"f!!"!""!+!
O!,!""!"!!c"!!!#"!+!
Q!,!"#!!!"`!!"!!!!"!+!
S!,!"!!!!"!!]"!!"!!!"!+!
U!,!"!!!"!!"Z!"!"!!!"!+!
W!,!"!!!"!"!X!"!"!!!"!+!
Y"+!"!!!"!"!V!"!"#"!+!
\!+!"#"!"!T!"!""#!+!
^!+!#""!"!R!"!!##!+!
`!+!##!!"!P!!"!!!!#!+!
b!+!#!!!!"!M"!!"!!!#!+!
d!+!#!!!!"!!K!"!"!!!#!+!
f!+!#!!!"!!"H!"!"!!!#!,!
f!,!#!!!"!"F!"!"!!!$!+!
h!+!$!!!"!"!D!"!"!"!#!+!
j!+!#!"!"!"!B!"!"!"!#!+!k!,!#!"!"!"!@!"
!#!!!#!,!
l!,!#!!!#!"!>!"!#!"!#!+!
n!+!#!"!#!"!<!#!"!"!#!+!
o!,!#!"!"!#!;!"!"!#!"!+!
q!,!"!#!"!"!:!"!#!"!"!,!
r!,!"!"!#!"!8!"!#!"!#!+!
s!,!#!"!#!"!6!#!"!#!"!+!
u!,!"!#!"!#!5!"!#!"!"!,!u!-
!"!"!#!"!4!"!#!"!#!+!
w!,!#!"!#!"!2!#!"!#!"!,!w!-
!"!#!"!#!0!#!"!#!#!+!
y!,!#!#!"!#!/!"!#!#!"!,!y!-
!"!#!#!"!.!"!#!#!#!+!
{!,!#!#!#!"!,!#!#!#!"!,!{!-
!"!#!#!#!*!#!#!#!#!+!E"!"!"!"w"!"!"!"!
!E!,!#!#!#!#)!#!#!#!"!,!B!!!.!!!p!!!.!!
!B!-!"!#!#!#!(!#!#!#!#!+!A!5"m!4!B!,!#!#
!#!#!'!#!#!#!"!,!@!8!j!8!@!-!"!#!#!#!&!#
!#!#!#!+!@!;!
$!?!,!#!#!#!#!%!#!#!#!"!,!?! $!=!>!-
!"!#!#!#!$!#!#!#!#!,!^!c!]!-
!#!#!#!#!#!#!#!#!"!-!=!A!
$!<!.!"!#!#!#!"!$!"!#!#!,!>!.!!!r!.!!!0!
=!-!#!#!"!$!!!#!#!#!#!,!J!!!%!.!_!,!!!%!
K!-!#!#!#!#!%!"!#!#!-!=!+!+!+!v!_!<!.!#!
#!"!$!#!#!#!#!-!<!+!w!+!9!;!.!#!#!#!#!#!
"!$!#!-!G!:!]!*!V!.!#!$!"!&!#!#!$!,!<!*!
<!f!<!;!-!$!#!#!%!#!#!$!,!<!)!y!)!=!;!-!
$!#!#!$!$!#!#!-!F!=![!)!Y!.!#!#!$!#!#!$!
#!-!<!)!=!e!=!;!.!#!$!#!#!#!$!#!,!=!)!y!
)!=!<!-!#!$!#!#!#!#!$!,!G!=![!)!Z!-!$!#!
#!"!$!#!#!-!>!(!<!f!<!=!.!#!#!$!!!#!$!#!
-!>!)!y!)!;!=!.!#!$!#!!!#!$!#!-!I!:!]!*!
X!.!#!$!#!!!#!$!1!?!*!8!j!8!>!2!$!#!%!#!
2!@!*!+!1!5!+!?!3!#!$!#!$!2!L!!!!!!!!!-!
a!+!!!!!!!!!M!3!$!#!#!$!2!A!?!
$!@!3!$!#!#!7!B!
$!=!A!8!#!#!7!_!9!!!!!!!!!!!!!!!!!!!!!9!
^!8!&!8!D!8!N!;!8!D!9!%!8!E"5!N!=!5"E!9!
>!H"/"P!@"/"G!X!K"("S!C"!!'!!!J!X!N!!"!"
T!I"!"N!X!
)!}!X!)!}!X! (!~!Y! &!~!Z! %!!Z!
%!!Z! $! !!Z! &![! $!\!
$!\! $!\! $!]! "!^! "!^!
"!^! "!_! !`! !a! }!b! }!c!
{!d! {!e! y!f! y!g! w!h! w!i!
u!j! u!k! s!l! s!m! q!o! o!q!
n!q! m!s! k!u! j!v! h!w! g!y!
f!z! d!|! b!~! `! !! ^! #! \!
%! Y" `!H!e!G! *! /!E! ,!D! .!
7!8!#!m!""7! :";"j!$!5" =!3!(!f"(!2!
@!1!,!a!,!0! B".!/"\"<"
E!,!2!!"R"!">! H"B"!"H"!!A"
K"E!!"!!;!!"!!D"
O"I!!"!"!"!"!"!"!"!!H" S" )" W"
%" [" !" "|" c"x" g$p$ m$h$
u$`$ }$X$ &%N% /&B& :&6&
F6

        g N
```

It may be hard for us to recognize, but to an RLE decoder program, this is a smiling face. Files such as the one above can be sent as parts of class-to-class electronic exchanges, or stored on computer bulletin board or conference systems as components of student graphics libraries as easily as text-only electronic mail is generated and received.

Do You Need a Captain Marvel Ring?

Encoding and decoding procedures are a lot less mysterious than they may sound. In CompuServe's "PICS Forum," software written for many different brands of personal computers can be downloaded and used to encode and decode pictures. All of the software stored in the PICS Forum is either freeware (legally copied and distributed for free) or shareware (legally copied and tested for free, but requiring a registration fee for continued use).

In addition, CompuServe maintains two very large online databases of encoded pictures that can be downloaded and decoded, then saved in binary picture file form, so that the downloaded graphics may be used in Logo programs. More complicated pictures (that take longer to download) are stored in the GALLERY Forum, and pictures that can be downloaded quickly are available from the QPICS Forum. Depending on modem speed, pictures can be downloaded in 4 - 15 minutes each.

Easy-to-use ProDOS RLE decoders for Apple II computers are stored in files called SAVRLE.EXE and RLEHNL.TXT in the PICS Forum. An Apple GIF decoder is available in the same place under the file name IIGIF.BNY. For IIgs users, a shareware program called SHRConvert can be downloaded as the file SHRCNV.BNY, then used to encode and decode both RLE and GIF pictures. The only program posted to date in the forum that will encode Apple II Logo pictures (saved with the SAVEPIC command) to RLE format is in a file called MRLE.EXE.

Want to Try it? Just ASCII!

Readers who do not have access to CompuServe or who are not comfortable with uploading and downloading procedures may send me a 5.25" diskette in a self-addressed, stamped diskette mailer. I will copy Apple II ProDOS RLE encoding and decoding programs onto the diskette, and enclose directions for use.

If you decide to browse in the PICS, GALLERY, and QPICS databases yourself, you may also want to visit the LOGOFORUM on CompuServe, where Logo-using educators from across the United States and Canada post classroom ideas, share Logo programs, and debate Logo philosophies. Any of the forums can be accessed by typing the command GO, followed by the name of the forum, at any ! prompt (e.g., GO LOGOFORUM). CompuServe subscription information can be obtained by calling their customer service department at 1-800- 848-8990 (in Ohio, 614-457-8650).

You've probably heard it said that one picture is worth a thousand words. In this case, a 256 X 192-pixel graphic is worth (at least) 350 text characters, which, in turn, can facilitate many new Logo connections.

Judi Harris
621F Madison Avenue
Charlottesville, VA 22903
CIS: 75116,1207
BitNet: JudiH@Virginia

Making Projects Personal: Reflections on a LEGO-Logo Workshop

by Mitchel Resnick
Learning and Epistemology Group
MIT Media Laboratory

During the past five years, I worked at many LEGO-Logo workshops. The workshops varied greatly. Some involved dozens of teachers, some involved only a few; some lasted a few hours, some lasted a few days. But in all of the workshops, one idea emerged again and again: the importance of *making projects personal*. Teachers who made *personal connections* with their projects invariably did the most creative work – and seemed to learn the most from the activity. In short, LEGO-Logo is at its best when people (be they teachers or students) care about their projects in a deep and personal way.

As an example, let me focus on one particular workshop: the Science and Whole Learning (SWL) workshop, organized last summer by Seymour Papert's Learning and Epistemology Group at the MIT Media Lab. The goal of this three-week workshop was to help teachers explore new ways of thinking about and learning about science. Roughly 60 teachers, mostly from Boston-area schools, attended the workshop.

LEGO-Logo played an important role at the SWL workshop. LEGO-Logo was viewed as one of two "basic tools" (along with LogoWriter) that teachers would be using throughout the workshop (and throughout the school year). The first three days of the workshop served as an introductory period for these basic tools. Each teacher spent two days working with and learning about LogoWriter and one day working with and learning about LEGO-Logo.

For the remaining 12 days of the workshop, teachers attended special-interest sessions and worked on their own personal projects. An entire room was set aside for teachers working on LEG-/Logo projects. Three of us from MIT (Steve Ocko, Natalie Rusk, and myself) supported the teachers as they worked on their projects. In addition, Eadie Adamson (a New York teacher and frequent contributor to *Logo Exchange*) and Cathy Helgoe (of LEGO Systems Inc.) offered a mini-workshop on "Kinetic Art" using LEGO-Logo.

Samples from a Workshop

Roughly a dozen teachers decided to make LEGO-Logo the focus of their personal projects. Here are a few examples:

Xyloman.

Carole Carter works as a music teacher at the Brimmer and May School, so it is not that surprising that she adopted a musical theme for her LEGO-Logo project. On the first day of the workshop, she decided that she would like to build a LEGO robot that plays the xylophone. The next day, she brought in a toy xylophone from home, and she began to build her robot. Eventually, Carole built a robot with two motors: one motor controlled the robot's arm, making it hit the keys on the xylophone; the other motor made the robot rotate so that it could hit different keys. She named the robot "Xyloman."

Next, Carole began writing computer programs to make Xyloman play sequences of notes. The programs used input from an optosensor to control the rotation of the robot. The sensing mechanism wasn't very accurate, so the robot had difficulties when successive notes were far apart. So Carole carefully chose songs in which the notes were close together. By the end of the workshop, the machine could play several different songs. Carole was clearly pleased by her machine's performance, but she added: "I haven't taught it how to bow yet."

Carousel.

Before the workshop, Natalie Isbitsky had never built anything out of LEGO bricks. But early in the workshop, she became obsessed with an idea: She wanted to build a LEGO carousel like the one she had ridden as a little girl at Revere Beach near Boston. Building a simple carousel proved relatively easy. But Natalie, a teacher at the Donald McKay School, wasn't satisfied. She wanted the horses to go up and down as the carousel rotated – just as they had on the carousel at Revere Beach. Natalie struggled with this idea, but she finally developed a mechanism that worked. She found a way to put a gear on the vertical axle so that the gear would remain stationary as the carousel turned. Then, she connected the horses to a set of gears positioned around (and at right angles to) the stationary central gear. As the carousel turned, these gears rotated around the stationary central gear – and made the horses go up and down.

Once the mechanism was working, Natalie wrote a computer program that played the song *And the Band Played*

On while the carousel rotated. She also decorated the carousel with some non-LEGO art materials – an idea she got from attending the Kinetic Art mini-workshop. At the end, Natalie reflected on her experiences: "Learning has always been a struggle for me. I'm amazed at how much I've learned during the past three weeks. At the start of the workshop, I didn't know how I would fill the time from 9 to 4. Now, there is never enough time in the day." She added: "My students are really reluctant to try new things. I can relate to that. I didn't know anything about Logo or music, and I felt uncomfortable about science. Now when I go back to school, I'll work differently with the students and give them more opportunities to try things. Letting the students do things hands-on will make them feel more engaged and make learning more personal."

Pole-balancing machine.

In the orientation session for the workshop, Seymour Papert led the teachers in an experiment. He asked the teachers to try to balance a yardstick vertically on their hands. Then he asked them to try to balance a pencil the same way. This experiment led to a discussion: why is a yardstick is easier to balance than a pencil? At one point, Seymour wondered aloud whether it would be possible to build a LEGO machine that could balance a pole on its "hand."

Kip Perkins of the Atrium School took up the challenge. He built a LEGO cart that moved along racks, and he placed a 1.5-meter wooden pole on top of the cart. He used just a single sensor for feedback: the sensor indicated which direction the pole was leaning. Kip "cheated" a bit to make the problem easier – he constrained the pole so that it could move in only one dimension, and he put a wing at the top of the pole to slow its motion. Still, many of us felt that Kip would not be able to make the machine work. We were wrong. By the end of the workshop, Kip's machine was reliably balancing the pole. His program was remarkably simple: it simply made the cart move (with a slight time delay) in the direction that the pole was leaning.

Fireworks machine.

The Sunday before the workshop began, Irene Hall went to watch the annual July 4 fireworks along the Charles River in Boston. So, when she sat down to work on a LogoWriter program on the first day of the workshop, she decided to draw fireworks on the screen. The next day, at the introductory LEGO-Logo workshop, Irene (who teaches at the Leonard School in Lawrence, Massachusetts) decided to continue with the same theme and build a fireworks machine.

Irene wasn't sure how to build a mechanism for throwing the fireworks, so she consulted the book *The Way Things Work* by David Macauley. She tried several different mechanisms: a spring, a cam, and finally a catapult. Irene used the catapult mechanism to throw glitter into the air – and ultimately onto a paper that had been covered with glue. The result was an interesting pattern of glitter glued to the paper. By the end of the workshop, Irene had developed a complete multimedia presentation. While the Logo turtle drew fireworks on the screen, the computer played the *William Tell Overture* and the LEGO machine threw glitter into the air. Said Irene: "I followed a path that just developed. That's why I learned so much. This exploration feels like everything learning should be about."

International drawbridge.

Ida Fallows, one of three Costa Rican teachers attending the workshop, had never seen a drawbridge before coming to Boston for the workshop. So Ida had no doubt what she wanted to do for her LEGO-Logo project. She looked through various LEGO building instructions until she found a mechanism that reminded her of the drawbridge. She built the mechanism according to the LEGO plans, then added some extra features to make it look more like the drawbridge she had seen.

When Ida was about to start programming her drawbridge, she discovered that another teacher, Steve Wachman of the Brimmer and May School, had been building a LEGO boat with motorized paddle wheels. So Ida and Steve decided to work together. Steve found a rolling cart in another lab and filled it with water. (As he put it: "I borrowed an ocean.") They put the boat and bridge into the water, added a light and optosensor, then wrote a program to coordinate the whole system. Whenever the boat passed in front of the optosensor, the drawbridge opened and gave the boat a chance to pass through. Finally, Ida added a song to the computer program: as the drawbridge opened, the computer played a Costa Rican children's song about a boat that couldn't navigate.

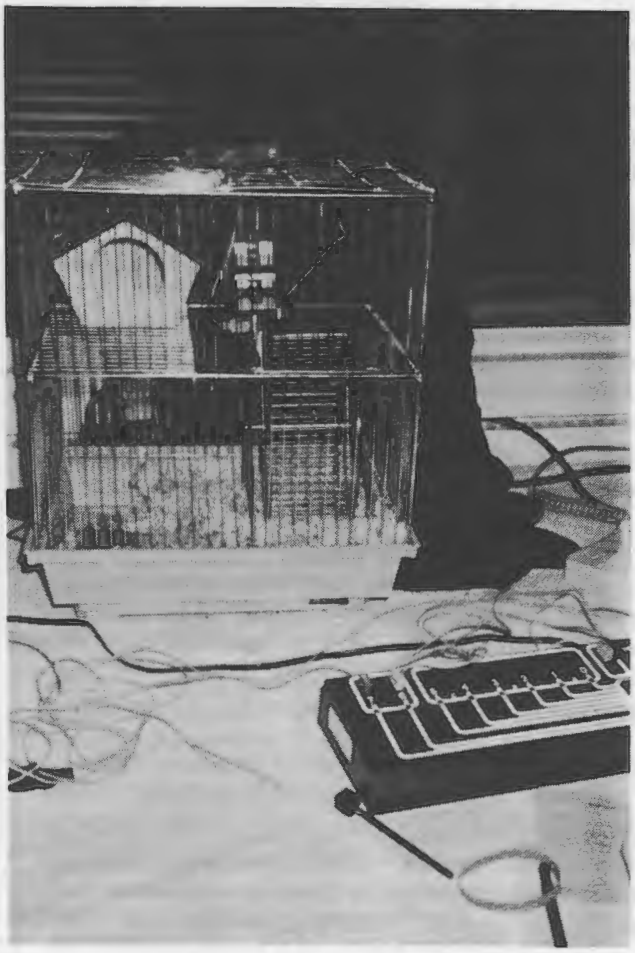
Mardi Gras.

The Kinetic Art workshop captured the imaginations of several teachers, including Sharon Beck of the Ellis School. Sharon wanted to build something that would evoke the carnival-like atmosphere of the annual Mardi Gras celebration in New Orleans. She combined LEGO motors and gears with a variety of non-LEGO materials, including pipe cleaners and flashy colored paper with optical patterns. One LEGO motor catapulted confetti into the air, while also controlling the dance of a pipe-cleaner turtle. A second motor spun a dazzling array of optical patterns. With both motors on, the LEGO machine turned into a lively and colorful work of art. Another teacher told Sharon: "We'll be thinking of you at Mardi Gras time."

LEGO-Logo zoo.

For a long time, Steve Ocko and I have talked about building a LEGO-Logo zoo, using LEGO sensors to monitor animal activity. During the workshop, we decided to give it a try. We bought a hamster and brought it into the lab. By coincidence, one of the teachers, Julie Fine of the Agassiz School, had also brought her hamster to the workshop. Julie planned to do a video analysis of her hamster running. After talking with us, she decided to work on the LEGO-Logo zoo as well. We attached optosensors to the exercise wheels in each of the hamster cages, and we wrote a program to keep track of how far (how many revolutions) each hamster ran every ten minutes.

The first day, the hamsters slept all day. The program reported a long list of zeros. We were pretty frustrated. But when we came back to the lab the next morning, the computer showed that there had been lots of activity overnight. One hamster ran several hundred revolutions between 8:00 and 10:00. The other hamster ran more than 1000 revolutions between 9:00 and midnight. We did a quick calculation and found that the hamster had run nearly a quarter mile! After midnight, both hamsters apparently went back to sleep. We continued to monitor the hamsters' activity for the next week. The hamsters ran exclusively at night, usually in two-hour bursts of activity. But our results weren't entirely reliable: several nights, one of the hamsters gnawed through the LEGO wires.



Cybernetic house.

At one of the tutorials during the workshop, I showed how analog sensors could be used in connection with LEGO-Logo. We borrowed some analog light and temperature sensors from the Technical Education Research Centers (TERC) in Cambridge. These sensors, which plug into the game port of the computer, can be read with the **paddle** command in the LEGO/Logo software. I suggested that someone might be interested in building a "cybernetic house" – that is, a house that regulates its own internal conditions.

Lego-Logo Workshop--continued

Dave Mellen, who teaches at the Kingwood Oxford School in Hartford, Connecticut, decided to work on the cybernetic house as his project. He and his son built a house out of LEGO bricks, and then Dave started adding sensors. He installed the TERC temperature sensor in the wall of the house, and built a fan to cool the house when necessary. The TERC light sensor wasn't responsive over the relevant range, so Dave bought a different light sensor at Radio Shack and added it to the house. Dave's program continually checked the two sensors and then took the necessary actions. If the light intensity dropped below a certain threshold, it would turn on LEGO lights inside the house. If the temperature rose, it would turn on the LEGO fan. Dave saw this as just the beginning: with more sensors and a more complex program, the house could behave almost like a living organism.

Reflections: The Role of Diversity

It is clear that these projects were rich in *personal meaning* for the teachers. For Costa Rican teacher Ida Fallows, who had recently seen her first drawbridge, the LEGO drawbridge symbolized her visit to the United States. For Natalie Isbitsky, the LEGO carousel symbolized her childhood. The carousel at Revere Beach, which Natalie had ridden as a child, is no longer there. Building a LEGO-Logo carousel was, in Natalie's words, "a way of regaining my youth."

What led to such a flourishing of personally-meaningful projects? The answer, I believe, revolves around the idea of *diversity*. The environment in our LEGO-Logo workshop encouraged and promoted diversity in several ways. The environment encouraged a diversity of project themes, a diversity of working styles, and a diversity of entry paths.

Diversity of project themes.

Question: What do a xylophone-playing robot, a sensor-monitored hamster cage, a multimedia fireworks display, and a self-regulating house have in common? Answer: Not much. But that is exactly the point. None of the projects in the workshop was "typical." Each was special in its own way.

To a certain extent, this diversity of project themes was a result of the tools we used. Both LEGO and Logo are open-ended construction sets: each can be used in many different ways. But tools alone do not lead to diversity. In organizing the workshop, we made diversity of projects an explicit goal. We didn't want the teachers to have preconceived notions of what LEGO-Logo projects *should* look like. We encouraged them to explore the boundaries of LEGO and Logo, to try to use the materials in new ways. Of course, we provided models and examples to give teachers a sense of what might be possible. We even suggested that, in the beginning, they might want to try some of the LEGO building instructions to get a "feel" for the LEGO materials. But we encouraged the teachers not to feel constrained in choosing themes for their longer-term projects. We wanted the teachers to be free to develop projects that were personally meaningful to them.

Diversity of working styles.

Different people work in different ways. Some people like to develop a plan then execute it; others like to tinker with the materials, letting a plan emerge as they work. Some people like to work in groups; others like to work alone. If a workshop (or a classroom) encourages and supports only some of these styles, some people will be left out. Only if a workshop respects and supports a diversity of working styles will participants feel comfortable enough to work on personally-meaningful projects.

Consider, for example, the question of collaborative vs. individual work. LEGO/Logo offers an ideal environment for children (or teachers) to work together as a team. In building a factory, each child can build an individual machine, then they can link them all together. Or perhaps some children can work on building the machines while other work on the programming. The value of collaboration was clear in the drawbridge project that Ida Fallows and Steve Wachman worked on together. And I have seen wonderful collaborative projects at other teacher workshops, particularly at the workshops organized by Gerry Kozberg and Paul Krocheski in St. Paul, Minnesota.

But sometimes, people need to work alone. Consider the case of Natalie Isbitsky. For the first several days of the SWL workshop, Natalie collaborated with another teacher in building a conveyor belt. That was a good learning experience, but Natalie didn't really get involved in the LEGO-Logo activity until she began working on the carousel. She felt much more of a personal stake in the carousel. She had a clear and compelling vision of what she wanted to do. It was *her* project.

Carole Carter went through a similar progression. From the very start of the workshop, she knew that she wanted to build a robot to play the xylophone. For the first few days, two other teachers worked with Carole on the project. The other teachers suggested that they put a rack on the xylophone, so that the xylophone could move back and forth. Carole preferred to make the robot, not the xylophone, move. (After all, that's the way it is in the real world.) Being a good team member, Carole agreed to put the xylophone on a rack, but inside she was feeling frustrated. After a few days, the other teachers lost interest in the project. Carole was relieved. She took the rack off of the xylophone, and continued with the project as she had originally envisioned it.

Diversity of entry paths.

The books accompanying the LEGO-Logo kit present a sequence of introductory activities involving LEGO cars and traffic lights. For some students and some teachers, these activities provide a good introduction to LEGO building and Logo programming. But these activities represent only one of many possible entry paths for LEGO-Logo. Cars and traffic lights do not appeal to all teachers or all students. Alternate entry paths are needed to capture the imaginations of other teachers and students.

The Kinetic Art mini-workshop, organized by Eadie Adamson and Cathy Helgoe, represents one alternative. The main goal of the Kinetic Art activity was to make teachers consider the creative and aesthetic aspects of building machines and structures. Eadie and Cathy brought a variety of art materials (colored papers, pipe cleaners, etc.) to the workshop, and they showed how these materials could be combined with LEGO mechanisms to create moving sculptures.

About a dozen teachers worked on some sort of Kinetic Art project. For several of these teachers, it was the first time that they felt a strong personal connection with LEGO-Logo. They had participated in other LEGO-Logo activities. They had built cars and traffic lights, but they hadn't felt a real personal involvement. The Kinetic Art activity resonated with these teachers. Suddenly, LEGO-Logo made sense to them. Now they understood why other people were excited about it. Finally, they had something that they really *wanted* to make.

The Kinetic Art activity could help address what some people see as a gender bias in the introductory LEGO/-ogo activities involving cars and traffic lights. Although both boys and girls have participated enthusiastically in these activities, some people worry that the boys feel a stronger personal involvement in the car-related activities. They worry that the girls, while participating, aren't as emotionally involved in the projects, and thus do not have as rich a learning experience.

Alternative entry paths for LEGO-Logo can help ease this problem. But developing good entry paths is not easy. One possible alternative is for children to begin by building LEGO houses. The children could use the computer to control the lights in the house. But this approach is not totally satisfying for me. Building houses seems like a watered-down approach: it misses out on the mechanical-design aspect of LEGO Technic.

Kinetic Art, by contrast, does not seem watered down at all. The teachers who built Kinetic Art sculptures at the SWL workshop were just as sophisticated in their use of LEGO materials as anyone else. In short, Kinetic Art seems like a excellent alternative approach for introducing LEGO-Logo. That is not to say that Kinetic Art is *the* right approach for everyone. There is no one "right approach." For the future, we need to develop more alternative paths, so that more teachers (and children) can make deep personal connections with LEGO-Logo.

Mitchel Resnick
Learning and Epistemology Group
MIT Media Laboratory

Statement of Ownership, Management and Circulation (Required by 39 U.S.C. 3685). 1A. Title of Publication, Logo Exchange. B. Publication No., 02789175. 2. Date of filing, October 16, 1989. 3. Frequency of issue, monthly except June, July, August. 3A. No. of issues published annually, 9. B. Annual membership dues, $25.00 4. Complete mailing address of known office of publication (not printer), ISTE, University of Oregon, 1787 Agate St., Eugene, OR 97403-9905. 5. Complete mailing address of the headquarters of general business offices of the publisher (not printer), ISTE, University of Oregon, 1787 Agate St., Eugene, OR 97403-9905. 6. Full names and complete mailing addresses of the publisher, editor, and managing editor, Publisher, ISTE; Editor, Sharon Yoder; Managing Editor, David Moursund; University of Oregon, 1787 Agate St., Eugene, OR 97403-9905. 7. Owner, International Society for Technology in Education, University of Oregon, 1787 Agate St., Eugene, OR 97403-9905. 8. Known bondholders, mortgages, and other security holders owning or holding one percent or more of total amount, None. 9. The purpose, function, and nonprofit status of this organization and the exempt status for Federal income tax purposes has not changed during preceding 12 months. 10. Extent and Nature of Circulation. Average No. Copies Each Issue During Preceding 12 Months. A. Total no. copies (net press run), 1,509. B1. Paid Circulation, sales through dealers and carriers, street vendors and counter sales, 0. B2. Paid Circulation, mail subscriptions, 1,230. C. Total Paid Circulation (sum of 10B1 and 10B2), 1,230. D. Free distribution by mail, carrier or other means, samples, complimentary, and other free copies, 200. E. Total Distribution (Sum of C and D), 1,430. F1. Copies not distributed, office use, left over, unaccounted, spoiled after printing, 79. F2. Copies not distributed, returns from news agents, 0. G. Total (Sum of E, F1 and 2—should be equal to net press run shown in A), 1,509. Actual No. Copies of Single Issue Published Nearest to Filing Date. A. Total no. copies (net press run): 1,535. B1. Paid Circulation, sales through dealers and carriers, street vendors and counter sales, 0. B2. Paid Circulation, mail subscriptions, 1,257. C. Total Paid Circulation (sum of 10B1 and 10B2), 1,257. D. Free distribution by mail, carrier or other means, samples, complimentary, and other free copies, 200. E. Total Distribution (Sum of C and D), 1,457. F1. Copies not distributed, office use, left over, unaccounted, spoiled after printing, 78. F2. Copies not distributed, returns from news agents, 0. G. Total (Sum of E, F1 and 2—should be equal to net press run shown in A), 1,535. 11. I certify that the statements made by me above are correct and complete. Tasanee Fry, Bookkeeper, ISTE.

Ireland – Logoland

The 1989 International Computer Problem Solving Contest: Senior Logo Results

by Donald T. Piele

The International Computer Problem Solving Contest (ICPSC) is an annual contest for precollege students conducted at local contest sites throughout the world. Held on the last Saturday in April, it challenges teams (from one to three members each) to solve five problems in two hours using any programming language. Last spring, approximately 300 teams participated in the Junior Logo Division (Grades 7-9). This month we present the results of the Senior Logo Division along with the problems and sample solutions. TheJunior Division was reported last month and the Elementary Logo division results will be presented in coming months. The BASIC and Pascal contest results appear in *The Computing Teacher*.

From 1981 to 1988, the contest was sponsored by the University of Wisconsin-Parkside. Currently it is sponsored by ICPSC – a non-profit organization with financial support from USENIX, the technical association of UNIX users.

Introduction

Since the Logo contest began in 1987, teams from contest sites in Ireland have won first place in at least one of the Logo Divisions of the ICPSC. This year they scored a double win – first place in the Junior and Senior Logo Division. This month the spotlight is on Andrew Farrell, the Senior Logo Division winner from Trinity College in Dublin, Ireland. Or is it Logoland?

"Andrew Farrell is a particularly interesting winner," writes Dr. Sean Close, lecturer in Mathematics Education at St. Patrick's College in Dublin.

> He is 14 years of age and had had almost no schooling when he first joined the St. Patrick's College centre early in 1986 at the age of 11. Whatever knowledge he possessed at that time he had acquired in his own home largely by self-study, but it was, in fact, a considerable body of knowledge. I arranged for him to take the SAT in May of that year and he achieved a total score of 1310 including a score or 760/800 on the SAT-Maths. As a result, he received counseling from Dr. Julian Stanley at John Hopkins University in Baltimore. In 1987 he achieved First Place in the Irish National Invitational Mathematics Contest, a contest involving the most able mathematics students in Irish secondary schools.

Having met its matriculation requirements in 1988, Andrew entered the University of Dublin, Trinity College as an undergraduate student of Mathematics. He attained First Class Honors at the end of his first academic year and shortly commences his second year there. Meanwhile he continued attending the extra-curricular courses in St. Patrick's College and has become a very skilled Logo programmer. He has a hard-disk MS-DOS PC at home which he programs in Basic, Logo, and C. His hobbies include computers, reading, tennis, cricket and puzzles. He has recently created a suite of puzzles which are to be produced commercially.

"Finally," Dr Close recounts,

Logo is very popular in Irish primary schools and, except for a few schools still using BASIC, is the only computer language taught. Logo is also gaining ground in the secondary schools where it is competing with BASIC, COMAL, and Pascal. The main

drawback to its development is the lack of hardware in the primary schools. But soon every school will have at least one PC with some of the bigger schools having up to 10 PCs.

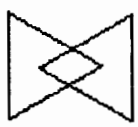
Dr. Close, the local ICPSC contest site director in Dublin, has been very influential in developing Logo courses for mathematically able students in Ireland. The children who attend his Logo courses are tested for a high proficiency in Mathematics and are aged from 9 to 12 years old. Among other things, these classes prepare the students for the Irish National Logo Contest, first held in Dublin in 1988. Children from all of the Logo centres in Ireland came together in June to participate.

Andrew was one of only three teams to solve all five problems in the ICPSC within the two hour time limit (out of the approximately fifty who tried). The average number of correct solutions was three.

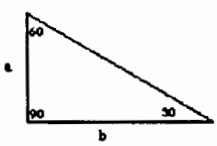
1989 Senior Logo Division Problems, Solutions

TWO TRIANGLES

Write a program to draw this design. Note that the point of each equilateral triangle is in the center of the other triangle.



Note that you may need the fact that for the triangle below, $a/b = 1/\sqrt{3} = .577$.



Solution

```
TO TWO.TRIANGLES
SET.UP
TRIANGLE
MOVE.TO.CENTER
TRIANGLE
END

TO SET.UP
CG
HT
END

TO TRIANGLE
REPEAT 3 [FORWARD 50 RIGHT 120]
END

TO MOVE.TO.CENTER
PENUP
RIGHT 30
FORWARD DIST.TO.CENTER
RIGHT 30
PENDOWN
END

TO DIST.TO.CENTER
OUTPUT SQRT ( 25 * 25) +
    (25 * .577 * 25 * .577)
END
```

Alternate DIST.TO.CENTER using trigonometry

```
TO DIST.TO.CENTER
OUTPUT 25/COS 30
END
```

DOWN UP

Write a program that accepts a word and prints it in a pattern like this:

```
DOWN.UP "PROBLEM
```

```
PROBLEM
 ROBLE
  OBL
   B
  OBL
 ROBLE
PROBLEM
```

```
DOWN.UP "SOLUTION
```

```
SOLUTION
 OLUTIO
  LUTI
   UT
  LUTI
 OLUTIO
SOLUTION
```

Test your program on the words "PROBLEM and "SOLUTION

Solution

```
TO DOWN.UP :WORD
DOWN :WORD 0
END

TO DOWN :WORD :COUNTER
REPEAT :COUNTER [TYPE CHAR 32]
PRINT :WORD
```

```
IF (COUNT :WORD)<3 [STOP]
IF NOT EMPTYP :WORD [DOWN BUTFIRST BUT-
   LAST :WORD :COUNTER +1]
REPEAT :COUNTER[TYPE CHAR 32]
PRINT :WORD
END
```

RANDOM SHAPES

Write a program that asks the user how many shapes s/he wants to see on the screen. The program should then pick a random spot on the screen and randomly choose a square, triangle or hexagon and draw it. For example, if you type

```
RANDOM.SHAPES
```

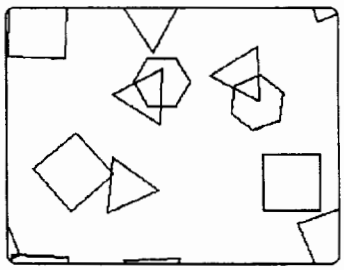
You see

```
HOW MANY SHAPES?
```

If you type

```
10
```

you see something like this:



Note that your square, triangle, and hexagon should be *approximately* the same size.

Solution

```
TO RANDOM.SHAPES
SET.UP
HOW.MANY
END

TO SET.UP
CG
CT
END

TO HOW.MANY
PRINT [HOW MANY SHAPES?]
REPEAT READWORD [PLACE SHAPE RANDOM 3]
END

TO PLACE
PENUP
FORWARD RANDOM 150
RIGHT RANDOM 360
PENDOWN
END

TO SHAPE :CHOICE
IF :CHOICE = 0 [SQUARE]
IF :CHOICE = 1 [TRIANGLE]
IF :CHOICE = 2 [HEXAGON]
END

TO SQUARE
REPEAT 4 [FORWARD 50 RIGHT 90]
END

TO TRIANGLE
REPEAT 3 [FORWARD 50 RIGHT 120]
END

TO HEXAGON
REPEAT 6 [FORWARD 25 RIGHT 60]
END
```

COUNT LIST

Write a program that accepts a list of numbers from the keyboard. It should then find the length of the longest sublist that contains just negatives. (A sublist of negatives consists of negative numbers immediately adjacent to one another.) That is, if you type

```
COUNT.LIST [1 3 8 -2 -9 -3 2 -9]
```

your program should print

```
LONGEST SUBLIST OF NEGATIVES IS 3
```

Test your program on the above list and the list `[1 -1 1 -1 1 -1 -1 -1 -1 1]`

Solution

```
TO COUNT.LIST :THE.LIST
FIND.NEGS :THE.LIST 0 0
END

TO FIND.NEGS :LIST :COUNT :LONGEST
IF EMPTYP :LIST [( PRINT [LONGEST SUBLIST
   OF NEGATIVES IS] :LONGEST ) STOP]
IFELSE ( FIRST :LIST ) < 0 [MAKE "COUNT
   :COUNT + 1] [MAKE "TEMP :COUNT]
IF :TEMP > :LONGEST [MAKE "LONGEST :TEMP
   MAKE "COUNT 0]
FIND.NEGS BUTFIRST :LIST :COUNT :LONGEST
END
```

COIN FLIPS

Write a program that simulates the flipping of a fair coin. Your program should ask the user how many coins s/he wants

to flip. The program should then graph each simulated flip as it occurs and should display the final totals when the simulation is complete. For example, if you type

```
COIN.FLIPS
```

you should see

```
HOW MANY?
```

If you type 100, the program might produce

```
# HEADS = 54 # TAILS = 46
```

Run your program twice with an input of 100.

Solution

```
TO COIN.FLIPS
SET.UP
ASK
END

TO SET.UP
CG
CT
END

TO ASK
PRINT [HOW MANY?]
FLIP.EM READWORD 0 0
END

TO FLIP.EM :NUMBER :HEADS :TAILS
IF :NUMBER = 0 [( PRINT [# HEADS =]
   :HEADS [# TAILS =] :TAILS ) STOP]
IFELSE ( RANDOM 2 ) = 0 [MAKE "HEADS
   :HEADS + 1 GRAPH.HEADS :HEADS]
   [MAKE "TAILS :TAILS + 1 GRAPH.TAILS
   :TAILS]
FLIP.EM :NUMBER - 1 :HEADS :TAILS
END

TO GRAPH.HEADS :NUMBER.HEADS
PU
SETPOS [0 0]
PD
REPEAT 2 [FORWARD :NUMBER.HEADS RIGHT 90
   FORWARD 20 RIGHT 90]
END

TO GRAPH.TAILS :NUMBER.TAILS
PU
SETPOS [30 0]
PD
REPEAT 2 [FORWARD :NUMBER.TAILS RIGHT 90
   FORWARD 20 RIGHT 90]
END
```

Conclusion

The major emphasis of the ICPSC contest is on the local contest site. Each year we receive many letters from local contest directors describing successful contests – some with awards ceremonies and banquets – where the winning team may have solved three problems. If we do our job right, everyone can solve some of the problems, and the very best in the world will be challenged to solve all five. The ICPSC provides the problems, sample solutions, instructions for the director and local judges–everything to make it easy for local teachers and administrators to conduct a successful contest. Our goal is simple – to promote the development of computer problem solving skills. Come join us!

1990 Contest

The 10th Annual ICPSC will be held on Saturday, April 28, 1990, with Friday April 27, and Monday, April 30 as alternate contest dates. For more information on how your school or school district can become a contest site, send your request to the address below. You will also receive a free copy of *Compute It!*, the newsletter of the ICPSC.

Donald T. Piele
ICPSC
P.O. Box 085664
Racine, WI 53408.

Senior Logo Division Rankings

Rank	Team	School	City, State/Country	Director	Advisor
First	Andrew Farrell	Trinity College	Dublin, Ireland	Dr. John S. Close	Dr. John S. Close
Second	Matt Arnold Al Go Paul Riddell	Live Oak H.S.	Morgan Hill, California	David Foster	David Foster
Third	Jens Larson	American School of Madrid	Madrid, Spain	J. Ernest Berlanger	

Logo and Company

Sight and Sound: The Next Decade Part II - Visions of the 90's

by Glen L. Bull, Gina L. Bull, and Chris Appert

The beginning of the decade seemed an appropriate time to examine classroom computing of the future. In last month's column we reviewed speech and sound capabilities. This month we will discuss images and visual media. Some of the capabilities which we will discuss are available for use in your classroom now. Other capabilities can be found in development laboratories, and may become available during the next decade.

The first versions of Logo had no graphics capabilities at all. Responses to commands were printed on rolls of paper at typewriter-like terminals. Experimentation with robotic floor turtles and access to graphic screens later led to development of turtle graphics. The now familiar commands of FORWARD, BACK, LEFT, and RIGHT provided an easy way to draw with the computer.

The motivation for development of turtle graphics lay in part with the concepts in mathematics and geometry that children could learn. Teachers and students embraced turtle graphics because it provided a way to produce drawings with the computer. The fundamental characteristic of Logo that distinguishes it from other computing languages is its extensible nature – the fact that you can add a new procedure to the language that behaves almost as though it were a built-in command. However, for many people, Logo and turtle graphics became almost indistinguishable. Turtle graphics played a key role in the rapid growth in the popularity of Logo during the late 1970's and early 1980's.

Paint and Illustration Programs

In theory children using Logo would find it necessary to explore mathematical concepts while using turtle graphics to draw pictures. At first Logo provided one of the easiest ways of creating computer graphics. During the latter half of the 1980's another creature, the mouse, changed the picture (in a manner of speaking). The introduction of MacPaint with release of the Macintosh popularized mouse-based paint and drawing programs. MacPaint was followed by programs such as PC-Paint for IBM-compatible computers and Mouse-Paint for Apple II computers.

Children who are familiar with paint programs may sometimes be less entranced by turtle graphics than they would be otherwise. Some types of pictures which require considerable programming skills in Logo can be sketched in a matter of moments with a sophisticated paint program. After a few minutes of experimentation the child can draw a picture that could require weeks of work in Logo. Of course, in the process the student may not be learning anything about angles or geometry or mathematics. From the child's perspective the end goal of completing the picture is achieved, however.

Many Apple II computers do not yet have mice, so paint programs are not an option for students who use these machines. Teachers with machines that do have mice might be inclined to conceal the capabilities of paint programs from students using Logo. However, in the long run it is better to integrate technological advances into the learning process rather than attempt to suppress them. Paint programs and Logo can be used in conjunction with one another. A background created using a paint program can be imported into many Logo programs such as LogoWriter, Apple Logo II, and Terrapin Logo through use of an *import* feature. An import command imports text or graphics created with another program into Logo. Graphics created with another program can be combined with the types of graphics for which Logo is best suited. For example,

- Logo is better suited to generation of certain types of geometric figures than paint programs are, and
- Logo also makes an ideal scripting program to generate animation and other visual effects through use of sprites.

Transfering a DOS 3.3 File to ProDOS

Graphics tablets such as Animation Station and the Koala Pad provide another means of creating pictures. This type of graphics tablet allows the user to sketch a drawing with a stylus. Both the graphics program which comes with the Animation Station graphics tablet and the one which comes the Koala Pad were developed for Apple II DOS 3.1. Therefore if you are using a version of Logo developed for ProDOS, such as LogoWriter 2.0, Terrapin LogoPlus, or LCSI Logo II for the Apple II, it will be necessary to convert the DOS 3.3 picture files to a ProDOS format. This is a straightforward process that can be accomplished with the ProDOS User's disk. This disk is provided with every Apple IIe, IIc, or IIgs computer sold, so if your copy is missing you should track it down or get a copy from your Apple dealer.

To give you an idea of the process, the steps involved in importing a picture created with Animation Station software to LogoWriter 2.0 format are listed below.

Transfer the Picture from DOS 3.3 to ProDOS Format

a. Place the ProDOS User's disk in Drive 1 of the Apple II. Place the DOS 3.3 disk containing the Animation Station picture in Drive 2. Then turn on the computer.

b. Select option "C" (Convert Files) on the ProDOS User's menu.

c. Remove the ProDOS User's disk and replace it with the LogoWriter 2.0 data disk.

d. Reset the ProDOS prefix for Drive 1.
   1. Select "P" (Set Prefix) on the convert menu.
   2. Then select "S" (by Slot & Drive) to reset the pathname.

   Choose Slot 6, Drive 1 by pressing the return key twice.

e. Transfer the DOS 3.3 file to the LogoWriter 2.0 disk by selecting "T" (for Transfer) from the convert menu. (Be sure that the direction shown at the top of the screen is:

   Direction: Dos 3.3 —> ProDOS

   Enter the file name of the DOS 3.3 picture file to be transferred. Animation Station adds the initials "PI" (picture) at the beginning of each filename. Therefore if you saved the picture of an automobile as "CAR", the file name will be: PI.CAR

Import the Picture into LogoWriter

a. Put the LogoWriter 2.0 startup disk in drive 1 and turn on the computer.

b. After LogoWriter has been loaded into the computer, replace the startup disk with your LogoWriter data disk, and press the ESC key.

c. Select the NEWPAGE option on the LogoWriter menu.

d. Load the picture into LogoWriter using the LOADPIC command. For example, if the name of the picture is PI.CAR, you would type LOADPIC "PI.CAR to import the picture into LogoWriter. The picture should appear on the LogoWriter screen.

   Note: If you forget the name of the picture, you can use the PRINT FILELIST commands to see a list of all the files (including non-LogoWriter files) on the LogoWriter data disk.

If you are currently using LogoWriter 1.0 or 1.1, you may want to upgrade to LogoWriter 2.0, since the earlier versions of LogoWriter do not have an import feature. Note that LogoWriter 2.0 requires 128K of memory. Some of the newer illustration programs such as Mousepaint use the ProDOS format. In the case of those programs, you can import the pictures directly without going through the steps of converting the picture from the DOS 3.3 to ProDOS file format.

Clip Art

Computer paint and drawing programs share one characteristic in common with other artistic mediums such as water colors and oil paintings. They are most effective in the hands of a person with artistic talent. It is possible to share the artistic wealth through *electronic clip art*. Clip art consists of illustrations developed by others. Some of this material is in the public domain; other is sold commercially. Clip art can be used to enhance an illustration drawn with a paint program. Thousands of clip art pictures are available for the Macintosh. Clip art quickly became available for IBM computers as well. In addition, there are translation programs to convert pictures created on the Macintosh to IBM formats. Such conversion programs may soon be available for the Apple II computer as well.

In the meantime there is a wealth of clip art which can be used with Logo programs on the Apple II as well. For example, a number of different Baudville clip art libraries are available which can be used directly with the Animation Station illustration program. There are also hundreds of public domain pictures created for the Print Shop format. A previous column by Judi Harris describes how pictures in the Print Shop format can be converted to a format that can be imported into Logo (*LOGOLINX*, SEPT. '88). Many teachers start a library of Logo art for their school, and save some of the most artistic pictures created by their students so that they can be used in later classes.

Image Digitizers

Video Digitizers

An image digitizer makes it possible to create your own clip art. As its name suggests, a video digitizer can capture an image from a videotape recording or a television camera, and convert it to a digital image that can be displayed on a computer screen. There are several inexpensive image digitizers for the Macintosh. One of the best known, MacVision, can be acquired through mail order houses for approximately $200. MacVision and MacRecorder (discussed last month)

Logo and Company--continued

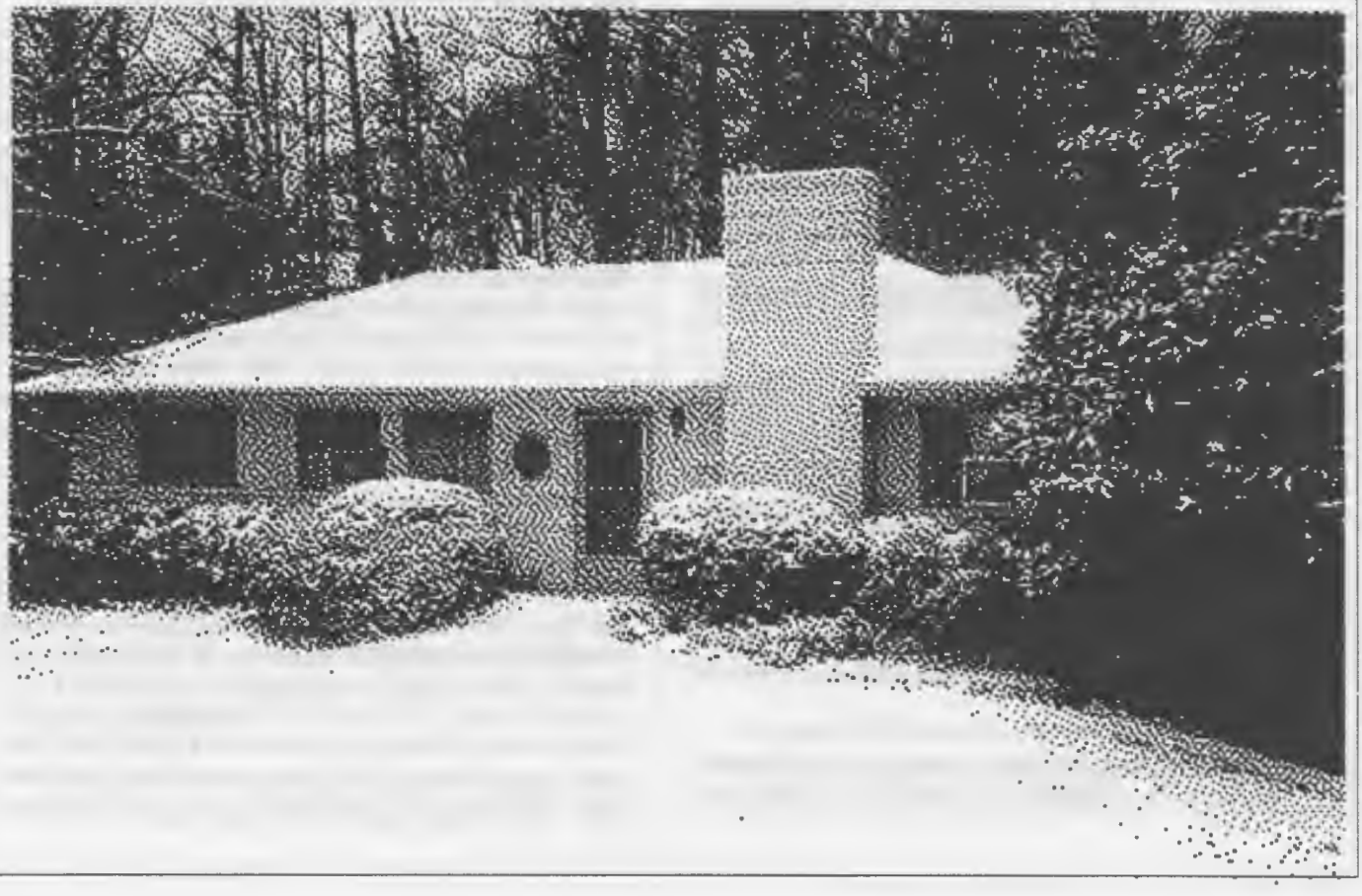
are a natural combination for any Macintosh computer. One digitizes images and the other digitizes sound.

Inexpensive image digitizers are also available for Apple II computers. Computer-Eyes is one that works well. The name itself makes us inclined to like the product. ("Computer-Eyes" can also be pronounced "computerize"; hence the name is a pun of sorts.) We recommend Version 2 of the software that accompanies Computer-Eyes. This version allows you to either acquire the image in an encrypted, packed format that takes up less room on disk, or in an unencoded format that can be imported directly into Logo. The image below of the Bull residence in winter is an example of a digitized picture.

Computer-Eyes captures images from any standard video source (a videocassette recorder, a video camera, etc.). The video source is connected to the Apple IIe or IIgs computer by means of an interface board which is placed in one of the Apple II slots. A phono plug to phono plug cable is used to connect the videocassette recorder (VCR) to the interface board. If you have access to a video camera as well as a VCR, you can connect the camera to the VCR in the normal way, and then connect the VCR to the Computer-Eyes interface board. We installed the system without difficulty using the clearly written instructions in the manual. The additional point worth noting is that it is important to plug the cable connecting the VCR and the Computer-Eyes interface board into the *video out* jack of the VCR.

The software which accompanies Computer-Eyes is also easy to use. The menu-driven program offers the user the option of using either the DOS 3.3 or ProDOS Apple II operating systems. If you will be using the Computer-Eyes pictures with Terrapin Logo select DOS 3.3. ProDOS would be the choice for work with Terrapin LogoPlus, LogoWriter 2.0, Apple Logo II, or LCSI Logo II for the Apple II. (If you are using LogoWriter 1.0, you will not be able to import pictures.) You will need a data disk formatted for use with the proper operating system (DOS 3.3 or ProDOS) to store your pictures.

The menus and documentation for the Computer-Eyes software will guide you through viewing and capturing an image. An "Adjust" option lets you control brightness and contrast in the image you are capturing. Various image types are available from "Hi-Res Single Level" to "Double Hi-Res Gray Scale." We found that using the Hi-Res Single Level, the simplest type, was appropriate for our purposes.



It takes about six seconds to capture an image, and it is essential that an image remain stationary during that time. If you are planning to capture images of people, especially children, using a video camera, it might be a good idea to caution your subjects to remain very still during the scan. It is often helpful to practice on inanimate objects first. Alternatively, you may want to use the video camera to record images on the VCR tape. Then you can use the VCR *pause* button to hold the image still while it is captured by the Computer-Eyes board. VCRs with four video heads are best for this purpose, but we found that by pressing the pause button several times we could often get a good pause (without static) even on VCRs with only two video heads.

After an image has been captured, you will want to save it to disk using the "Disk Access" menu. The program offers a choice of saving pictures in a packed or unpacked format. Select the *unpacked* format for pictures which will be used in Logo programs. The saved pictures can be loaded directly into Logo using import commands such as LOADPIC in LogoWriter 2.0.

Scanners

A scanner looks much like a copy machine. A flat sheet of paper is placed on the glass surface of the scanner. However, instead of creating a second copy on a sheet of paper, the scanner digitizes the image. A scanner works much like image digitizers such as MacRecorder or Computer-Eyes. However, it may be more convenient for two dimensional objects. Some scanners can only digitize graphics, others only digitize text, while some can digitize both if the appropriate software is available. A wide variety of scanners are available for Macintosh and IBM-compatible computers. We do not know of any scanners with interface cards for Apple II computers.

Analog Images

All of the images discussed above are digital ones. Video images have more colors and higher resolution than most computer pictures. However, mu*ltimedia* techniques make it possible to combine computer and video technologies.

Videodisc Players

A single Apple II data disk will store 10 to 20 pictures in an unpacked format. It is possible to store a few more pictures by saving them in an encrypted packed format that saves space, but the pictures have to be unpacked before they can be used with Logo. In contrast, more than 50,000 slides can be stored on each side of a videodisc. A videodisc plays back video images on a television monitor in much the same way that a videocassette player does. However, the images are stored on a laser disc that resembles a compact music disk. This makes it possible to go directly to any one of the 50,000 images.

Many videodisc players such as the Pioneer 4200 or the Sony 1200 have a serial port that makes it possible to connect them directly to the computer. Programs such as Hypercard on the Macintosh or Linkway on the IBM can be used to control the videodisc player. On the Apple II, programs such as Tutor-Tech, Hyper Studio, or HyperScreen can be used to control the videodisc. It is also possible to send commands to the videodisc player with many versions of Logo.

Sending Commands to the Videodisc Player

Once the videodisc player has been attached to the computer, a short procedure Send is added to Logo. **Send** sends the text out the serial port, where it is received by the videodisc player. (Readers of last month's column will recognize that **Send** is simply a renamed **Say** command.) For example, if the videodisc player is attached to a serial card in Slot 1 of an Apple II computer, the **Send** procedure would be written in the following way in Apple Logo II. (The DRIBBLE command sends text printed on the screen to the specified file or device.)

```
TO Send :Command
DRIBBLE 1
PRINT  :Command
NODRIBBLE
END
```

Various commands are sent to the videodisc player using the send command. These commands differ slightly for each brand of videodisc player. For example, the letters "PL" represent the play command for Pioneer 4200 videodisc player. Therefore, a play procedure for this videodisc player might look like this in Logo:

```
To Play
Send [PL]
END
```

This allows students using Logo to send commands of this type to the videodisc player:

```
FIND SHARK
FIND BUTTERFLY
FIND AMOEBA
```

With discs such as the BioSci disc (available from Video Discovery), which contains more than 20,000 plants and animals, students have a wide variety of images to choose from.

Videodiscs and LogoWriter

A final note about videodiscs and Logo. Unfortunately the command for sending text out the serial port is not documented for LogoWriter. However, the support staff at Logo Computer Systems, Inc (LCSI) was very helpful and generous with their time in showing us how to do this, even though this command is not documented in the LogoWriter manual.

This allowed us to create our own set of LogoWriter commands such as FIND and PLAY which we use to control the videodisc player. LCSI has also created a disk of LogoWriter commands which they have developed as well. If you have a videodisc player such as the Pioneer 4200 which you wish to wish to control with LogoWriter, you may wish to talk with LCSI.

Videocassette Recorders

More recently, an interface card which can be used to control a VCR player with an Apple II computer has just been announced. The disadvantage of a VCR is that it is necessary to go through the entire tape to get to a picture at the end. Even with fast forward this sometimes takes awhile. On the other hand, there are many more VCRs in the public schools than there are videodisc players. A number of years ago we had an interface card for controlling VCRs, but it was somewhat awkward and clumsy to use and we soon discarded it. Since we have not yet acquired one of the new VCR controllers we do not know whether it can be easily interfaced with Logo. However, a new generation of consumer VCRs which allow the user to specify an exact minute and second on the tape, either directly or through computer control, will almost certainly appear sometime during the next decade.

Video Overlay Cards

In the applications that we described involving videodisc players and videocassette recorders, the video picture would typically appear in a separate video monitor sitting beside the computer monitor. Apple has introduced an inexpensive video overlay card which allow the computer image and the video picture to be displayed on the same screen. We have acquired an Apple II video overlay card, and hope to report more about its potential use with Logo and other programs in future columns.

Overview of Graphic Tools for Logo

Obviously Turtle graphics are the traditional means of creating graphics in Logo. For some traditionalists, it will remain the only means. However, for the rest of us, during the next decade a wide range of tools will become available for creating and displaying graphic images with Logo.

Method	Features and Capabilities
Paint Programs	Easy to Learn
Graphics Tablet	Uses Pencil-Like Stylus
Clip Art	Supplements Paint Programs
Video Digitizer	Captures Three-Dimensional Images
Scanner	Captures Printed Images
Videodisc	50,000 Analog Pictures
VCR	Widely Available
Video Overlay Card	Combines Computer & Video

Back to the Future

This two-part series on "dimensions of sight and sound" has reviewed some of the technologies which may find their way into the classroom during the next decade. Many are already available. Other technologies which are not even on the horizon are sure to give us realistic graphics and sound before the decade is over.

Many of these technologies can be used with Logo. Other programs have also been developed for utilization of these emerging multimedia capabilities. These applications include programs such as Hypercard on the Macintosh and Linkway on the IBM. On the Apple II applications such as HyperScreen, Hyper Studio, Tutor-Tech, Slide Shop, and VCR Companion have been developed to allow teachers to take advantage of these technologies.

Earlier in the year we asked for nominations for the best descriptor for this genre of "Logo-like" applications. Our best definition of a "Logo-like" tool is that it gives the learner control of the technology. Susan Jo Russell coined the term "learner-based tool" to described this category of applications, and we liked the phrase so much that we have written a couple of articles which incorporate this term in the title. However, the *Journal of Learner-Based Tools* has a rather formal ring.

Liz Wiener of Lawrence, New York, sent us the suggested nominations that we like best in response to our contest for the best descriptor for "Logo-like" applications. She suggested "Logo-ware" or possibly "Logoistics". In return for the best suggestion we received, we will send her one of the

best visions of Logo, James Clayson's *Visual Modeling with Logo: A Structured Approach to Seeing*, from the Exploring with Logo series edited by Paul Goldenberg. (If you would like your own copy, and we highly recommend it, you can obtain it from M.I.T. Press. Ask your local bookseller to order a copy ... or two ... or three.)

The limitation of all the nominations we received or thought of ourselves for the class of "Logo-like" applications is that they all contain the term "Logo". While this may not seem to be a disadvantage to Logophiles such as ourselves, it is apt to be perceived as chauvinistic by others. We suspect that someone who first was introduced to the genre through a program such as Hypercard or Hyper Studio would prefer a more neutral term. Of course, we may be incorrect in our assumption that programs in this genre all have a commonality. It is possible that those using Logo would say it has nothing in common with Hypercard or HyperScreen, or vice versa. However, we enjoy all of these programs, and think otherwise. (In fairness, we should point out that one of the authors, Gina Bull, thinks that Unix and Logo share a number of common features, such as extensibility.)

What do you think? In the coming decade of the nineties should the *Logo Exchange* journal incorporate discussions of "Logo-like" applications such as HyperScreen and Hyper Studio, or should it confine itself to Logo and dialects of Logo such as LogoWriter? What other applications would you like to see explored in the *Logo Exchange*? Send your opinions and suggestions to us or to the editor, Sharon Yoder. If we receive a sufficient number of comments and points of view, we will continue this discussion in a future column.

Glen Bull is a member of the instructional technology faculty in the Curry School of Education at the University of Virginia. Gina Bull is a programmer analyst for the University of Virginia Department of Computer Science. By day she works in a Unix environment; by night, in a Logo environment. Chris Appert teaches at the Kluge Children's Rehabilitation Center, University of Virginia, and uses computers with children of all ages.

Glen and Gina Bull
Curry School of Education
Ruffner Hall
University of Virginia
Charlottesville, VA 22903

BITNET addresses:
Glen: LB2B@ VIRGINIA. Gina: RLBOP@VIRGINIA.

ICPSC--A Problem Solving Challenge

What is the International Computer Problem Solving Contest (ICPSC)?
To some schools it is a chance to challenge their top computer and mathematics students in an annual computer problem solving event that compares their solutions to the best in the world. To others it is challenging set of problems to be used as enrichment material for computer programming classes. However you choose to use the ICPSC, it can be a valuable resource for any computer programming teacher.

What is unique about the ICPSC?
The ICPSC combines the art of problem solving with the skill of computer programming. Our contest challenges teams (from one to three students each) to create short, original solutions to a set of five problems within a two-hour period. All problems can be solved with short computer programs written in any language on any computer system. The problems fall into five categories: 1) computation, 2) simulation, 3) graphic patterns, 4)words, and 5) mind benders.

Can an individual teacher in a school become a contest director for his/her school?
Yes! We want to challenge and enrich your students so you don't have to make it a major district-wide event if that would stop you from doing it. Of course the more students in your local area who can get involved the better. But, if you only have one student in your whole school who wants to compete, that is all you need.

Can Logo be used?
Our original contest problems were not appropriate for Logo students. So we added a new Logo contest and last year, we expanded this contest to include all three age groups.

What are all the age groups?
Elementary Division (Grades 4-6); Junior Division (Grades 7-9); Senior Division (Grades 10-12); These are now offered in either the Open contest (all languages) or the Logo contest.

When is the contest held?
The 9th Annual ICPSC is scheduled for Saturday, April 29, 1989. Friday, April 28 and Monday, May 1, are alternate dates that can be used if Saturday is impossible. This year's set of contest problems will be mailed to the contest director on April 1, 1989. Sets of previous problems and solutions for students to practice on are always available.

How can I get more information about the ICPSC?
Dr. Donald T. Piele, ICPSC, P.O. Box 085664
Racine, WI 53408, Ph 414-634-0868

Logo: Search and Research

To err is human...to debug, divine
by Douglas H. Clements

Considering the many Logo programmings bugs we have seen in the last several columns, it would be easy to despair. But there are three good reasons to resist such negativism.

1. All bugs are not created equal(ly)

Some species of Logo bugs are rare. In an analysis of over 2,400 error messages from 19,000 Logo commands, du Boulay reported that 11 errors accounted for 96% of the total errors. These errors included undefined procedures (28%), insufficient arguments (16%), and extraneous text (10%). The students' most persistent errors were concerned with variables.

Thus, the most pestilent misconceptions should definitely be understood by the teacher and addressed instructionally with care. However, this involves but a few basic misconceptions. Several were described in previous columns. A major one is described next.

2. Many bugs arise from a common source

Pea (1986) claims that several persistent pests share a common connection. They result from widespread conceptual misunderstandings that arise when students learn to give instructions to a computer. That is, they allow their behavior to be guided by an analogy to conversing with another person. For example, consider the following three bugs.

Parallelism bug

Students with this bug believe that different lines in a program can be active, or known to the computer, in parallel. For example, many high school students believe that an IF/THEN statement such as IF :SIZE = 10 THEN PRINT "HELLO remains active. So, anytime during a program that the condition becomes true, they believe that HELLO will be printed. This reflects the influence of natural language, in that statements such as "if you have to sharpen your pencil, then..." do not just apply for an instant. Usually they are in effect anytime one wishes a more pointed writing instrument. Often these students believe that there is an intelligence "beneath the surface" of the Logo code which monitors the entire program: "It looks at the program all at once because it is so fast" (p. 28)

Intentionality bugs

Here, students attribute goal directedness or foresightedness to the program. For example, students may see one statement that reminds them of a procedure they have seen and remark, "it wants to draw a square." As with the parallelism bug, the student confers on the program the characteristic of an intentional being that is aware of its goals and its internal operation.

Egocentrism bugs

These bugs are the flip side of the page. Here, students believe that there is more meaning regarding their own goals in their programs than is actually present in their code. They believe that they can omit lines because the computer "knows" what they want it to do, and can "fill in the gaps" as human listeners often do. This belief is implicit only, but it is responsible for many errors, such as omitting punctuation or commands.

These three bugs are related. According to Pea, they are derived from one "superbug." In other words, the locus of Logo locusts is often the belief that there is a hidden mind somewhere in the language that has intelligent, interpretive powers. Students may not believe this explicitly, but they act as if they do. Their default strategy for making sense of of programming problems is to use analogies from natural language conversation. This should not be viewed by teachers as abnormal behavior, but should be expected. In fact, it is widely observed (Kuspa & Sleeman, 1985).

Classroom implications

Pea offers several suggestions for teaching. First, we need to be aware of the widespread misconceptions arising from analogies to conversation. That is, we need to help students debug their mental models and beliefs about computers in general and Logo in particular. Second, we have to arrange more diagnostic and learning opportunities that will make these bugs palpable. The bugs live on when they are not confronted. Reading and interpreting Logo programs, in individual, diagnostic, and group settings, should be a frequent activity. So should hand-tracing programming to debug :hem. Students must come to learn what needs to be made explicit and what does not. They also need to learn about how a language handles the flow of control, possibly through the presentation of clear models, explicit think–aloud examples of how good programmers think about programs, and instruction in monitoring one's comprehension of programs.

3. Logo bugs should not be exterminated, but cultured

Finally, we must remember that many Logo bugs are helpful critters, if seen as an invitation and an opportunity to learn. Teachers should promote a "learning from debugging" atmosphere. Bugs in procedures should be welcomed, not avoided at all costs. They can lead children to reflect on their own thinking processes and reveal misconceptions. Of course, this has always been a tenet of the Logo philosophy.

> Many children are held back in their learning because they have a model of learning in which you have either "got it" or "got it wrong." But when you learn to program a computer you almost never get it right the first time. Learning to be a master programmer is learning to become highly skilled at isolating and correcting "bugs...". If this way of looking at intellectual products were generalized to how the larger culture thinks about knowledge and its acquisition, we all might be less intimidated by our fears of "being wrong." This potential influence of the computer on changing our notion of a black and white version of our successes and failures is an example of using the computer as an "object-to-think-with..." thinking about learning by analogy with developing a program is a powerful and accessible way to get started on becoming more articulate about one's debugging strategies and more deliberate about improving them. (Papert, 1980, p. 23)
>
> Errors benefit us because they lead us to study what happened, to understand what went wrong, and through understanding, to fix it. Experience with computer programming leads children more effectively than any other activity to "believe in" debugging. (p. 114)

But only, we might add, if we accept this philosophy in our teaching. The persistence of certain is probably due in no small part to a lack of explication and articulation. We need to facilitate students' reflection and discussion about their bugs and their debugging. It should be noted this approach differs fundamentally from recent suggestions to teach of specific, behavioral "steps to debugging."

References

Kuspa, L., & Sleeman, D. (1985). *Novice Logo errors.* Unpublished manuscript, Stanford University, Stanford, CA.

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

Pea, R. D. (1986). Language–independent conceptual "bugs" in novice programming. *Journal of Educational Computing Research, 2,* 25-36.

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

Global News

Paradise Lost, Gained and Lost Again

Edited by Dennis Harper

The Caribbean is in many ways a paradise – tropical breezes, beautiful water, lush flora, and colorful fish swimming among large coral beds. On the other hand – islands stretch for 3000 miles and lingua francas are numerous making communications difficult; crime and drug usage are increasing, bureaucracies and a laid-back lifestyle often stifle innovation, and the only turtles most people have ever heard about are the ones in the azure seas.

In my six years as international editor of the *Logo Exchange*, I have heard of only one small Logo project (a 1987 Jamaica study) taking place in any of the 31 nations of the Caribbean. Upon arrival in the Virgin Islands, I did not meet one faculty member at the University of the Virgin Island (UVI) or in the VI schools who was even modestly familiar with Logo. I found myself in virgin territory.

In January of 1989, UVI began offering a master's degree program in computers in education; we included a course with a heavy Logo emphasis. The course was given during last August and was unique in many ways:

- The course took place on three different islands (St. Thomas, St. Croix, and St. John). Participants first moved from island to island and the last week of the course was given via distance learning.

- Twenty-one experienced American Logo teachers joined students from around the Caribbean.

- To help in the instruction of all these 51 students, five well known Logo instructors came to the Caribbean – Tom Lough, Glen Bull, George Uhlig, Mary Anne Gillis from the U.S. and Iliana Nikolova from Bulgaria. These 27 experienced Logo users provided the Caribbean beginners with a powerful foundation. In addition, Judi Harris talked to the group via telecommunications links.

- Both Lego®TC Logo and LCSI provided tremendous support through the loan of 10 Lego/Logo kits and LogoWriter for all participants. I would like to thank both these companies for their valuable assistance.

- Although this was an immersion course in Logo, participants still had time to take a sailing trip on an old

schooner, snorkel on coral reefs, see the sights, and enjoy the beaches and their resort hotels.

The Logo experiences and good will generated by this experience will not soon be forgotten by the participants. If this sounds like a 1980 Logo testimonial, you're right. One of the benefits of introducing Logo to a country or region is that one can continually relive the early days and witness a new lot of teachers getting excited about Logo. It is even better now than it was ten years ago; LCSI and Terrapin's Logo versions are much better now and also there is Lego®TC/Logo!

Since last August, three master's degree candidates are now doing Logo research, Tom Lough sold eight Lego/Logo sets in the Caribbean, and one BASIC teacher is converting to Logo this semester.

Now back to the title of this column. . . . With more than one hundred teachers taking graduate level computer education courses on four islands, numerous other countries wanting to enter these courses, Logo taking off, 23 schools joining telecommunications networks such as MIX, Star Schools, Kidsnet, Campus 2000, and AGE, and the Caribbean forming an ISTE organizational chapter, it looked as though paradise was being gained again. Then came Sept. 17-18, 1989.

Hurricane Hugo hit with a vengeance! The island of St. Croix which only last month had been the site of much Logo activity, experienced near complete devastation. St. Thomas, St. John, St. Kitts, Nevis, Montserrat, and St. Maarten were hit extremely hard. These are all areas are serviced by UVI. With no electricity for weeks or months, torn down computer labs, water-damaged software and school closures, it could be some time before any program will continue. Most trees have been blown away, shrubberies have lost their flowers and leaves and residents are now in a survival mode.

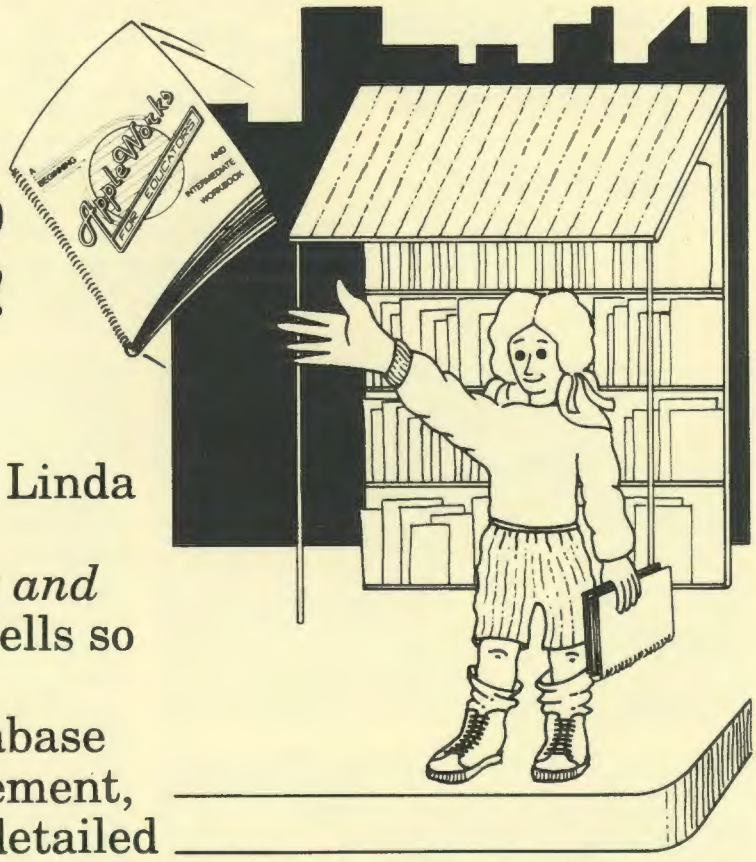
Even with the resumption of electricity and replacement of damaged materials (luckily, most of UVI's computers were safe but the public schools lost many) priority for spending will be on rebuilding the physical plant. Gone now are the monies allocated for our Lego/Logo kits, Terrapin Macintosh Logo, video disk players, telecommunications, scanner, etc. But the Caribbean will come back, stronger than ever. This has been a temporary setback as the seeds to a better education for Caribbean residents have been planted and there is no turning back now.

Educators are now trying to determine how technology can help rebuild this devastated region. Some ideas are using SAT software to help students prepare for their exams as they have missed so much school already, designing data bases to track school damage and repair, and the use of distance learning for students who have no school. I'll keep you posted on further developments.

This is the last international column of the 1980s. The *Logo Exchange* looks forward to reporting on the many exciting Logo activities that will be taking place around the globe in the 1990s.

Before Hugo: Tom Lough, Glen Bull, and Dennis Harper enjoy the ocean

AppleWorks for Educators—A Beginning and Intermediate Workbook hits ISTE's best-seller list.



Over 20,000 copies sold!

There's a good reason Linda Rathje's *AppleWorks for Educators—A Beginning and Intermediate Workbook* sells so well. It works.

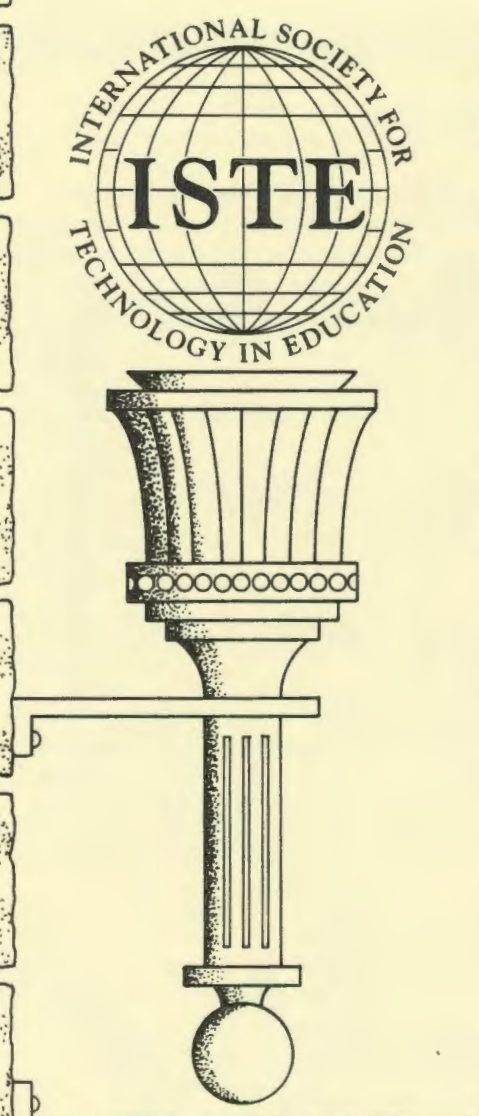
Word processing, database and spreadsheet management, and printer options are detailed step-by-step. Both novice and experienced AppleWorks users benefit from the depth and strength of the material.

AppleWorks for Educators—A Beginning and Intermediate Workbook has been revised to include a mail merge section, expanded integration activities, glossary, and up-to-date articles from *The Computing Teacher*.

Move *AppleWorks for Educators—A Beginning and Intermediate Workbook* to the top of your reading list.

$22.95

ISTE, University of Oregon, 1787 Agate St., Eugene, OR, 97403-9905; ph. 686-4414.



The ***International Society for Technology in Education*** touches all corners of the world. As the largest international non-profit professional organization serving computer using educators, we are dedicated to the improvement of education through the use and integration of technology.

Drawing from the resources of committed professionals worldwide, ISTE provides information that is always up-to-date, compelling, and relevant to your educational responsibilities.

Periodicals, books and courseware, *Special Interest Groups*, *Independent Study* courses, professional committees, and the Private Sector Council all strive to help enhance the quality of information you receive.

Rely on ISTE support:

- *The Computing Teacher* draws on active and creative K-12 educators to provide feature articles and carefully selected columns.
- The *Update* newsletter reaches members with information on the activities of ISTE and its affiliates.
- The *Journal of Research on Computing in Education* comes out with articles on original research project descriptions and evaluations, the state of the art, and theoretical essays that define and extend the field of educational computing.
- Books and courseware enhance teaching materials for K-12 and higher education.
- Professional Committees develop and monitor policy statements on software use, ethics, preview centers, and legislative action.
- The Private Sector Council promotes cooperation between educational technology professionals, manufacturers, publishers, and other private sector organizations.

It's a big world, but with the joint efforts of educators like yourself, ISTE brings it closer. Be a part of the international sharing of educational ideas and technology. Join ISTE.

Join today, and discover how ISTE puts you in touch with the world.

Basic one year membership includes eight issues each of the *Update* newsletter and *The Computing Teacher*, full voting privileges, and a 10% discount off ISTE books and courseware. $28.50

Professional one year membership includes eight issues each of the*Update* newsletter and *The Computing Teacher*, four issues of the *Journal of Research on Computing in Education*, full voting privileges, and a 10% discount off ISTE books and courseware. $55.00

ISTE, University of Oregon,
1787 Agate St., Eugene, OR 97403-9905.
ph. 503/686-4414.