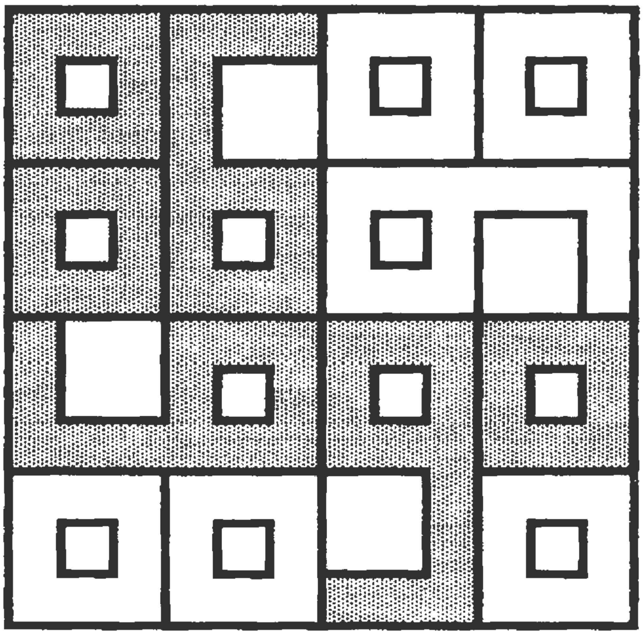
LOGO 86

LOGO 86 PROCEEDING

Massachusetts Institute of Technology
Cambridge, Massachusetts
9-11 July 1986

SPONSORED BY:
Epistemology and Learning Group
M I T
20 Ames Street E15-309
Cambridge, MA 02139



Created by:

Mark Frydenberg

LOGO 86 Steering Committee:

Harold Abelson
Richard Carter
Brian Harvey
William Higginson
Jacqueline Karaaslanian
Thomas Lough
Seymour Papert
Joyce Tobias

Welcome to Logo 86

Logo 86 will have three plenary sessions and nine parallel sessions. The main purpose of these Pre-Proceedings is to help you plan your choice of parallel sessions. On the following pages you will find the conference timetable, a sessions-at-a-glance chart, indices of the papers by session and by author, and the papers themselves.

In each of the parallel session time slots there are six concurrent sessions. The parallel session time slots are numbered: for example, "parallel session 3" is the time slot from 4:30 to 5:30 Wednesday afternoon. Each of the six concurrent sessions is identified by a letter between A and F. The sessions are *very roughly* organized into strands as follows:

A. Elementary education

B. Teacher training and miscellaneous

C. Secondary education

D. New technology

E. Research

F. Meta-Logo (talking about Logo)

However, you should not restrict your attention on the basis of this rough division because sometimes one strand overflows into another. For example, session C-4 (Lego/Logo) is really more relevant to elementary than to secondary education, but session A-4 was already taken with another elementary education topic.

A more reliable guide to the overall thrust of a session is the *mini-stream*, if any, to which it belongs; the sessions-at-a-glance chart shows these mini-streams as vertical bars and labels. For example, sessions C-7, C-8, and C-9 form the Secondary Mathematics mini-stream. Almost all of the papers presented at Logo 86 are part of some mini-stream. Sessions A-9 and E-9 contain papers that didn't seem to fit in any of the usual categories; I hope you will consider these sessions despite the lack of overall theme, because some of the papers seem quite interesting.

In planning a conference like this one, there is always a conflict between the goal of providing an opportunity for as many people as possible to present their work and the goal of keeping the conference experience manageable for the people who attend. We have taken several steps this year to try to balance these goals:

• At Logo 85 there were 10 or 11 concurrent presentations in each time slot. The ideal would be to present two or three of the most outstanding papers only. This year we are compromising by limiting the number of concurrent presentations to six.

• To allow presentation even of work with a limited plausible audience, we are experimenting with an increased use of poster sessions. There were "poster" sessions at Logo 84 that actually involved audio-visual demonstrations, in an inadequate, traffic-jammed, smoke-filled space. This year we are presenting about 50 true poster sessions, in a much larger space. You should find time to explore the poster area and perhaps get in touch with people whose work is relevant to your own.

• We are reducing the presentation time for most papers from 45 minutes to 30 minutes. It seemed to the reviewers that a great many papers presented an important but narrowly focused idea, and that a short presentation would be appropriate. We think that most presentations will benefit from the smaller time slots, although a few may be cramped. Certain presentations have been scheduled for 45-minute slots because the reviewers found those papers either particularly impressive or particularly broad in scope. A very few papers were scheduled for one-hour slots, not because they are even better than the 45-minute papers but because of some special characteristic: the session is a panel discussion with several presenters; it is a tutorial introduction to a complicated topic; or the session will involve extensive demonstrations or audience participation.

Last year, like everyone else, I hated the fact that two papers were combined into one session. I swore to myself that I wasn't going to allow that to happen this time, but in fact this year it's even worse: there are 90-minute sessions containing *three* 30-minute presentations. The trouble is that a schedule with a single presentation per session eats up half the day in passing periods! I've tried very hard to organize sessions so that the presentations within each session really will appeal to the same audiences. Also, to solve one of the problems from last year, we are insisting that presenters make their presentations in the order listed in the program.

I would like to offer my thanks to the paper reviewers, mostly students and staff at the MIT Media Laboratory; to the other members of the Logo 86 steering committee; to everyone who submitted a paper; and specifically to Sylvia Weir, who is not listed as a member of the steering committee but who contributed a good deal of her time and good advice when I was in the final stages of arranging the program.

Brian Harvey
Program Chair

P.S. Thank you for not smoking in the session rooms, the exhibit and poster areas, the Kresge and Athletic Center lobbies, and other indoor areas.

Parallel Sessions at a Glance

	Mezzanine Lounge (elementary) A	West Lounge (training & misc.) B	Room 407 (secondary) C	Rehearsal Room A (technology) D	Room 491 (research) E	Rehearsal Room B (meta-logo) F
Wednesday 10:45 1	LISTS FOR KIDS Swan — Primarily Rosen — But Wait HO	Hillel* — w/o FD & SK Skillen — Empowering HO	SECONDARY APPLICATIONS OUTSIDE MATH Eisenberg — Spider Hemingway — Dance Stremikis — Weather	LOGO EXTENSIONS Pinkteren — AIBLE Haars — Structure Edit Squires — Developing	RESEARCH METHODOLOGY Norton — Literacy Hopmann — Document Olive — Dribble	META-NOTATION Goldenberg — Iconic Bouchard — Necklace Sandys — Diagram
3:00 2	Headlight*	MICROWORLDS Templar — Creativity HO Feurzeig — Intelligent HO	Harroff — Language Lough — Physics	LeGallais — Curriculum Tempel — Beyond Turtle	Watt — Collaborating Wilder — Evaluation	ADVANCED LOGO TEXTS Burke — Formal Harvey — Projects
4:30 3	Headlight*	Cohen — Fostering Watts — Developmental	INTERNATIONAL Bordeleau — Quebec Rodriguez — Spain	SPEECH Birch — Puppet Theater Cartmell — Three	Burrowes — Success Livesay — Parents	Birch — Playing with Minsky — Data Rep. HO
Thursday 8:30 4	INSTANT Cochran — Instant Logo Scherer — Preschoolers	TEACHER TRAINING Solomon — Teachers Handler — Curriculum	Ocko* — Lego Logo	LOGO WITH OTHER SOFTWARE Garrigan — PostScript Goldberg — TeleLogo	COGNITION Clements — Metacog Jaworski — Transform	McClees — panel: Teachers' perspect.
10:00 5	ELEMENTARY MATH Stager — If There Had Carter — Algebra Neufeld — Learning	Schaffer — Thinking Brucker — Approach Colgan — Writing	Goldenberg — Math Systems Frydenberg — Turtle Grows	TOOLS IN LOGO Fournier — MICREL Wasser — ProLogo Wegman — Disc. Nets	Harel — Kids as McDougall — Structure Adams — Microworlds	MINI-TUTORIALS Davidson* — Syntax Silverman — Tools
4:00 6	Ary — Fractions Kliman — Fractions Winkler — Integers	Tempel — Staff Chesebrough — Skill Charischak — Living	LISTS AND RECURSION Denenberg — Two Logos Lewis — Lists, Data Sandys — List Games	Guadial — Sea of Text Abduleser — w/o Prog Roffman — Knowl. Rep.	AFFECT Clements — Social Garrigan — Robots	Dawson* — Logo w/o Silverman — Mules
Friday 8:30 7	LOGO IN THE CURRICULUM Wolff — Learning Templar — Authentically	MUSIC Upitis* — Music	SECONDARY MATH Binswanger — Agnes Irwin Reisch — Senior Math	OBJECT PROGRAMMING Davidson — Intro	SPECIAL ED Bull — Network Bernstein — Special	STYLE Burrowes — Contradict Kliman — Styles
10:00 8	Leventhal — Total Goldowsky — Global Weinberg — K-5 Manual	Fry — Musicland Moran — Music Logo Stager — Logo & Music	Colgan — Logo and Math Bouley — Linear Algebra Bolotin — Summermath	Jones — What Good Bergeron — Building Ringleberg — Turtles	Nevile — Learning From Wols — Teaching Pascal	Reggini* — Artisanal Watt — Logo in China
4:00 9	Bonnier — Individuality Schenk — In Touch Carver — Playing	CROSS-CULTURAL Sower — Native Amer. Gray — Romancing Gill — Logo Culture	Mugrage — Calculus Kern — Turtle Geometry Kline — 3D Turtle	LINGUISTICS Goldenberg* — Language Friendly — Language	Cunningham — Girls Carmichael — Dimensions Robbins — Texas	PHILOSOPHY Donnelly — Logoism Robertson — Getting McCurdy — Shouldn't

* Kresge Little Theatre

Conference Timetable

	WEDNESDAY		THURSDAY	FRIDAY
8:45	Plenary Session 1 Seymour Papert et al. "Logo Philosophy Revisited in a Real Logo-Based School"	8:30	Parallel Session 4	Parallel Session 7
		9:30	Coffee	Coffee
		10:00	Parallel Session 5	Parallel Session 8
10:15	Coffee			
10:45	Parallel Session 1 rehearsal B Rm 491			
11:15		11:45	Sponsor Session 3	Sponsor Session 5
12:15	Lunch	12:15	Lunch	Lunch
1:40	Sponsor Session 1	1:45	Sponsor Session 4	Sponsor Session 6
2:20	Sponsor Session 2	2:30	Plenary Session 2 B. Harvey, W. Higginson "Whatever Happened to the Revolution?"	Plenary Session 3 Burrowes, Goldenberg, McCauley Panel: "Cognition vs. Curriculum"
3:00	Parallel Session 2 rehearsal B			
		3:30	Coffee	Coffee
4:00	Coffee	4:00	Parallel Session 6	Parallel Session 9
4:30	Parallel Session 3			
5:30	West lounge	5:30		

LOGO 86

DINING AND SPECIAL EVENTS

Tuesday, 8 July

In conjunction with Registration there will be a Welcoming Reception on Tuesday evening from 6:00 - 8:00 pm in the Sala de Puerto Rico. All conference participants and their guests are invited.

Wednesday, 9 July

The Conference Banquet will be held on Wednesday evening beginning at 6:00 pm and will feature a traditional New England Clambake in an outdoor setting. The menu includes fish chowder, steamed clams, boiled Maine lobster, barbequed chicken, corn bread, watermelon, beer, wine and soft drinks. Following the clambake, live entertainment will begin at 9:00 pm inside of Walker Memorial. Tickets may be purchased at the Registration/Information Desk.

Thursday, 10 July

On Thursday evening the Sponsors and Exhibitors will host a reception from 6:00 - 7:30 pm in the Courtyard of McCormick Hall. All participants and their guests are invited to attend.

LUNCHES

LUNCH will be served Wednesday through Friday in the Athletics Center. The cost of these meals is included in the registration fee. Participants with dietary restrictions should alert the Information Desk as soon as possible.

COFFEE BREAKS

Refreshments will be available mid-morning and mid-afternoon in the Sala de Puerto Rico, located on the second floor of the MIT Student Center.

Logo86 Papers by Session

Parallel Session 1: Wednesday, 10:45–12:15

Strand A: Lists for Kids

Strand B

Strand C: Secondary Applications Outside Math

Strand D: Logo Extensions

Logo86 Papers by Session

Strand E: Research Methodology

Strand F: Meta-Notation

Logo86 Papers by Session

Parallel Session 2: Wednesday, 3:00–4:00

Strand A

Strand B: Microworlds

Strand C: Secondary Applications Outside Math

Strand D: Logo Extensions (LogoWriter)

Strand E: Research Methodology

Strand F: Advanced Logo Texts

Logo86 Papers by Session

Parallel Session 3: Wednesday, 4:30–5:30

Strand A

Strand B: Microworlds

Strand C: International Reports

Strand D: Speech Synthesis

Strand E: Research Methodology

Logo86 Papers by Session

Strand F: Advanced Logo Texts

Parallel Session 4: Thursday, 8:30–9:30

Strand A: Instant Logo

Strand B: Teacher Training

Strand C

Strand D: Logo with Other Software

Strand E: Cognition

Strand F

Logo86 Papers by Session

Parallel Session 5: Thursday, 10:00–11:30

Strand A: Elementary Math

Strand B: Teacher Training

Strand C

Strand D: Tools in Logo

Logo86 Papers by Session

Strand E: Cognition

Strand F: Mini-Tutorials

Parallel Session 6: Thursday, 4:00–5:30

Strand A: Elementary Math

Strand B: Teacher Training

Strand C: Teaching about Lists and Recursion

Strand D: Tools in Logo

Strand E: Affect

Strand F

Parallel Session 7: Friday, 8:30–9:30

Strand A: Logo in the Elementary Curriculum

Strand B: Music

Strand C: Secondary Math

Strand D: Object Programming

Strand E: Special Education

Strand F: Programming Style

Logo86 Papers by Session

Parallel Session 8: Friday, 10:00–11:30

Strand A: Logo in the Elementary Curriculum

Strand B: Music

Strand C: Secondary Math

Strand D: Object Programming

Logo86 Papers by Session

Strand E

Strand F

Parallel Session 9: Friday, 4:00–5:30

Strand A

Strand B: Cross-Cultural

Strand C: Secondary Math

Strand D: Linguistics

Strand E

Strand F: Philosophy

Logo86 Papers by Author

Logo86 Papers by Author

Logo86 Papers by Author

Logo86 Papers by Author

Logo86 Papers by Author

Logo86 Papers by Author

LOGO AS TROJAN HORSE: RETHINKING LOGO PHILOSOPHY IN THE CONTEXT OF A REAL SCHOOL EXPERIENCE.

Seymour Papert
Massachusetts Institute of Technology
Cambridge, MA

It is appropriate at the opening plenary session to take stock of where the Logo movement is this year and to propose themes for thought and discussion that will resonate with its current concerns.

Two years ago at LOGO84 the dominant mood was exuberance: the simple fact that Logo was becoming a reality in schools was still exciting in itself. The dominant intellectual interests were those most closely related to getting started. At LOGO85 there was more questioning of the meaning of this reality. In the plenary sessions and in informal discussion more attention was paid to criticism from without and from within.

I hope that exuberance and self-criticism will again be present this year. But I sense a gear shift. The focus of interest has shifted from getting started and justifying Logo to integrating it into the life of a school. How can Logo interact with the curriculum and with the learning style in the school?

Of course these issues have been present in discussion about Logo from the beginning. But I believe that the time is ripe for a stronger drive towards deeper understanding and more effective action. There is a well established broad base of Logo practioners in schools. We have more concrete examples on which to base discussion of Logo and the curriculum. And I feel that theoretical ideas relevant to the issues have become sharper. In my address I shall draw on the past year's experience in the Hennigan School to describe some new examples and to review old and new theoretical issues.

Although the situation in this school is exceptional in many ways, including a very high density of computers, examples of work developed there are applicable in situations with more modest resources.

The chief theoretical issue I shall talk about touches on two themes that will be explored more deeply in the other plenary sessions: "Curriculum vs Cognition" and "Reform vs Revolution." The central question for educators is whether schools of the future will go on teaching the same curriculum, using computers to do the job better, or whether we'll see radical change in what is taught and what is learned in schools. In my address I shall suggest that the education system will not be able to bring itself to decide on radical change in the curriculum. But there will be radical change all the same, despite the system's lack of decision making power. Change will come about through what I call "The Trojan Horse Effect" -- the system will admit what look like innocuous materials which will serve as vehicles for "germs" of change.

Of course the analogy with Troy is not exact. Our agents of change are not enemies -- everyone will be better off in the end. And there need be no deception if the the "horse" is so valuable even from the traditional perspective that it is seen as an offer they can't refuse.

Logo is not the only Trojan Horse on the scene. But I do think that Logo-based methods will be the most powerful Trojan Horses and that the Logo community is by disposition as well as by expertise the set of people most likely to carry out this kind of strategy and thereby take its place as the foremost agent of change in the contemporary world of education.

PRIMARILY LISTS

Karen Swan
Intelligent Learning Systems Laboratory
Teachers College, Columbia University

It has been suggested that there are two Logos -- the "no threshold" Logo of elementary turtle graphics and the "no ceiling" Logo of advanced list processing. Grounded in Bruner's thesis that anything can be taught at any level, this presentation will argue that the concepts of list processing are among the most powerful "powerful ideas" embedded in the Logo language; that Logo without these is an emasculated Logo; and that list processing not only can, but should be introduced to primary students. The integration of list processing with turtle graphics in primary instruction, methods for using lists to introduce other powerful ideas such as variables and recursion, and activities and materials successfully used with first, second and third graders will be discussed.

BUT, WAIT...THERE'S MORE

Marian B Rosen
Ladue Public Schools & Webster University
St. Louis, Missouri

An important goal of Logo is to have people explore, question, control, and master an environment. Students working on a self-defined project can then coincidentally be brought into contact with powerful ideas in math and programming. Logo teaching should center on the teacher's ability to help learners define projects consonant with their abilities and age levels, and then to design mini-lessons and/or mini-tasks that help them to reach their goals.

Most teachers, even those who do allow students to define their own projects, begin with simple turtle graphics. Occasionally students ask to make their grahpics interactive in some way. "Can I fix it so it will ask what size house to draw? "Wouldn't it be neat if I could drive the turtle down the road?" But usually they do not -- probably because they are not used to working with a medium as creative as Logo nor to dreaming of the possible. Occasionally teachers suggest that programs could be made interactive. But usually they do not -- probably because they think of turtle graphics and lists as two "things" separated by a chasm of difficulty.

In fact there are ways to use graphics and lists at every level of Logo. The dividing line is not your FORWARD vs. your BUTFIRST. There is no necessary dividing line. There is a continuum of increasingly complex projects in all areas.

All of graphics is not "easy". For example, since the Logo language is itself a simulation of Mathland, it is possible to draw circles based on Turtle, Euclidian, or Cartesian geometry. Students enter "graphics" at levels appropriate to their interest, knowledge and ability. Drawing a single-key-stroke square is turtle grahpics. But so is the exploration of curvature and topology.

All of lists is not "hard". First graders can print short stories to go with their drawings. Third graders can break their drawings down into discrete parts and ask a user to indicate which part they want to have drawn next. (This, incidentally, is a nice way to encourage modular progamming.) Fifth graders can get a computer to ask for somebody's name and use it in a sentence. These activities are all excursions into lists. (Students working with sprites/ multiple turtles find it particularly easy to begin using lists since it is natural to NAME them individually or by groups. The new LogoWriter will, at last, make multiple turtles available to the majority of Logo users.)

If Logo teachers show examples and explain that it is possible to add interactive elements to a graphics program, students begin thinking of ways to turn their pictures into games. As these games grow, they require answer-checking, score-keeping, direction-giving, etc. Writing such procedures involves the use of ever more sophisticated list work while maintaining the motivation that comes with working on a self-defined project. Furthermore, the process of making programs interactive, no matter in how simple or complex a manner, is empowering for students. They are introduced to the principles used to write commercially prepared software. That which was mysterious becomes accessible. That which was the province of experts becomes territory for them to explore.

Committed to this approach to Logo, I created the Sixth Grade Idea-Lab Software Project. Three groups of five, ten, and eight intellectually-gifted sixth graders were scheduled to spend twenty-three minutes a day in the computer•lab with me and twelve units of Sprite Logo. The childrens' knowledge of Logo ranged from nil to thirty hours of work in summer Logo Camps. I presented them with the challenge of defining something they thought young children should learn and then planning a computer game that would teach it. I showed them examples of Logo games written by others. I demonstrated their tool kit -- turtles that could draw, turtles that could act like sprites, commands that would print words on the screen and collect information from a user, ways to test that information and make decisions based on it. I promised them that I would protect them from failure by helping them to choose projects I felt sure they could master and by writing "tools" for them if necessary. They were encouraged to work in teams, although eight rugged individualists chose to go it alone. They would also be required to write documentation for their games stating the educational objective, age level, and directions for playing.

Opportunities to introduce list commands arose naturally from the nature of the project. E.g., "I want to be able to answer Yes, Yeah, or Y" led to a discussion of the FIRST of a word. "I want it so the answer can be "bird", "birds", "chicken", or "chickens" led to MEMBERP. "I'm tired of typing this riddle program over and over. Isn't there some way to just change the last line?" led to inputting a whole list of words as a variable. "I want to write a fourth level to this game but there is no more memory!" inspired a search for a master procedure that took a dozen inputs. "Isn't there some way to have the computer pick the numbers in the math problems?" led to naming random outputs.

The visual results of this project are charming. There are counting games in which the user drives the turtle around the screen with an interactive single-stroke program connecting numbered dots and drawing a picture. There are several matching games in which the turtle "picks up" objects and puts them in a designated place. Reading games demand that users drive their turtle to specified shapes, words, and numbers. A beginning consonant game demands to know the names of various objects in a classroom and if the user guesses correctly, the object is drawn on the screen. There are addition, subtraction, and multiplication drills as well as a spelling drill in which the reward is a runner rounding bases to score for his team. There are riddles and even a program that allows the user to stamp shapes on the screen and thus create pictures. Most of the projects have rewards and some of them keep track of scores or give hints if the user gets stuck. The games will actually be used in various elementary school computer labs around the St. Louis area.

The educational results of the software-project were contact with the powerful ideas involved in interactive programming and the empowerment that comes from turning an idea into a concrete, playable game. Only a few of the students could independently program games of the same sophistication next month. But all of them understand that they can collect input from a user, name it, manipulate it, compare it to other stored information, and write a series of IF...THEN statements that cause the computer to respond. A few of the students were extrememly frustrated at times, but all of them evidenced self-confidence and pride in both their product and their programming ability at the end of the project.

The difficulty with implementing game type Logo projects has been that the teachers must know about list and language manipulation and these topics have generally been seen as being "advanced". The new LogoWriter will help to change this attitude in that it offers not only multiple turtles, but also easier ways to use text and the elimination of work space problems. There are, however, still implications for teacher training. It is important that grahpics and lists be introduced together so that a false division is not implied and inferred. It is important that meta-Logo concepts such as naming, variables, recursion and testing be emphasized and recognized in more than one guise, in lists as well as in graphics. It is important that the project approach to Logo be modeled so that teachers will pass it on to their students.

The advantage of working with list handling primitives is that it can actually make teaching Logo easier. Most teachers run out of ideas for using one turtle. They get caught in the trap of "pushing" geometry and, thereby sometimes kill interest in Logo. Multiple turtles and interactive commands open whole new worlds for exploration. The intent of this presentation is to show a few of the possible directions this exploration can take.

FD, BK, LT, RT are good, but WAIT,...THERE'S MORE.

For further information, please contact:
Marian B Rosen, 7541 Marillac Drive, St. Louis, Missouri 63133

TURTLE-GEOMETRY WITHOUT FD AND BK: AN ATTEMPT TO SABOTAGE THE 'DRAWING-SCHEMA'

J. Hillel, C. Kieran, S. Erlwanger, L. Winer(*)

Young children (age 8-12) are generally introduced to LOGO through the 'drawing with the turtle' metaphor. The success of this metaphor is attested by the evidence of a strong drawing schema underlying the children's behaviour. Consistent with this schema, children's goals are those of drawing a particular figure in some specific screen location. Figures tend to be viewed as composed of linear or curvilinear segments and their productions generally proceed from the exterior boundary towards the interior, with a strong preference to avoid retracing already drawn lines. The planning strategies are local ('planning-in-action') and 'more-or-less' solutions are perfectly acceptable.

There are no conflicts within the drawing-schema if a procedural definition of a figure is identified with a particular screen output. In a sense, attributing a name to a production serves precisely to allow one to forget the details of the actual process (from the child's point of view). Conflicts within the drawing-schema arise when children are introduced to some of the procedural mechanisms such as iteration, calling subprocedures, variable and recursion. For one thing, exploiting these techniques calls for a global rather than a local analysis of geometric figures. It also calls for a dynamic rather than a static representation of procedures.

There is ample research evidence (e.g.[1], [2]) that children, even those with a fair amount of turtle-geometric experience, spontaneously revert to the drawing-schema when faced with the task of creating a procedural description for a complex figure. There is a reluctance, for example,

(*) Research supported by Quebec Ministry of Education, FCAC Grant #3004. C. Kieran is at Université du Québec à Montréal. The other authors are at Concordia University.

to define subprocedures or to retrace the same line segment, particularly when figures are connected or superimposed. The preponderance of 'interface' bugs also suggests cognitive difficulties in the representation of procedures.

Our reseach looks at alternate ways of introducing turtle-geometry so as to 'sabotage' the drawing-schema. Thus children initially work with a LOGO 'environment' which contains only the following 'primitives':

1. MOVE

   The turtle moves X steps in the direction of its heading or in the opposite direction (depending on whether X>0 or X<0) without leaving a trace. The turtle moves slowly.

2. TRT and TLT

   The turtle turns right and left but slowly enough to show an obvious rotation.

3. TEE

   The turtle produces a letter T with input signifying the length of the line segments. The production is state-transparent.

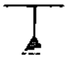
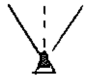
4. VEE

   The turtle produces a letter V with input signifying the length of the arms. The constructed angle is 60° and the production is state-transparent. VEE was modified in Session 8 to NEW.VEE in which 60° was generalized to an arbitrary angle by adding a second variable.

5. REPEAT, HT, ST, CS, EDIT and related commands.

Comments

(i) The 'environment' was designed to exclude the possibility of producing line segments (at least till the children found a clever way of exploiting NEW.VEE!). Thus any complex figure had to be described in modular terms using TEE, VEE or other children-defined procedures.

(ii) The productions of and were deliberately made state-transparent (the turtle's initial and final state for TEE was  and for VEE  , i.e. for VEE the heading was on the bisector of the angle). It was hoped that this would enable the researchers to discuss the notion of transparency once the children defined their own procedures (which are seldom transparent).

(iii) The 'primitives' were deliberately introduced with inputs to foster the idea of 'generalized' procedures. (See [3].)

(iv) It was hoped that providing the children with ready-made descriptions of objects would help the awareness that the location of an object on the computer screen can be solved independently of its description as a particular LOGO production.

THE ACTUAL RESEARCH

The above 'environment' was introduced to 3 pairs of novice 12 year olds for a total of 10 weeks (one hour per week). For the first 5 sessions, the children worked exclusively on tasks initiated by the researchers. Tasks were chosen with the aim of provoking the children to reflect on the invariants of objects such as TEE and VEE, on the utility of procedures and on relations involving angles. In particular, exact solutions involving the figure required working with appropriate multiples of 30° rather than the predominant multiples-of-45° scheme which is often sufficient for 'more or less' solutions.

The data obtained (via video-taping the screen's output, dribble-files and observational notes) were analyzed from several perspectives:

1. Knowledge of angles and of the multiple relations among the turtle's heading, the direction and angle of rotation and the constructed angle

of a figure.

2. The definition and use of procedures.

3. Awareness of turtle-state.

4. The nature of the children's solutions: Are the choices for inputs to the commands based on the analysis of the task or simply on perceptual cues (i.e. "this line looks like about a RT 60").

The same group of children are now starting a 12-week session with 'normal' turtle-geometry. Our intention is to trace influences from their initial experience which have become operational in their new work. In particular we would like to see if, and to what extent, the ideas of modularity and state-transparency are used or whether the children will simply revert to a predominantly drawing schema.

References

[1] Hillel,J. & Samurçay, R., (1985), Analysis of a LOGO environment for learning the concept of procedures with variable, Research Report #2, Concordia University

[2] Mendelsohn, P., (1985), L'analyse psychologique des activités de programmation chez l'enfant de CM^1 et CM^2, ENFANCE, No.2-3

[3] Report from Working Group on Variable in Proceedings of the LOGO and Mathematics Education Conference (C.Hoyles and R.Noss, eds.), University of London, Institute of Education, 1985

EMPOWERING CHILDREN

by

Peter A. Skillen

This paper reflects a Janusian approach in that I am merging two distinct and (often) opposing branches of psychology in using Logo with elementary school children. In synthesizing (humanistic) intrinsic motivational psychology with (generally behaviourist) cognitive science research, my wish is to **empower the learner** - to give control of the learning to the child. Practical classroom applications are given to support this vision.

Do we want our students to be '**origins**' or '**pawns**' ? DeCharms (1971, p381-382) describes the difference. (The pronoun 'he' is used in the original work. One can assume that this is gender-free.)

> "An Origin is a person who feels that he is the director of his life. He feels that what he is doing is the result of his own free choice; he is doing it because he wants to do it, and the consequences of his activity will be valuable to him. He thinks carefully about what he wants in this world, now and in the future, and chooses the most important goals, ruling out those that are for him too easy or too risky. He goes about determining what he can do concretely *now* to help himself toward his goals. His obvious assurance is not bravado, for he is not rash. Rather, he is genuinely *self-confident* because he hsa determined how to reach his goals through his own efforts. He is not 'Dependent of Destiny' since he is aware of his abilities and limitations. Relying on personal insight gleaned from instructive failures and skillful successes, he contrives to allow his strengths to outweigh his weaknesses. In short, an Origin is master of his own fate...
>
> A Pawn is a person who feels that someone, or something else, is in control of his fate. He feels that what he is doing has been imposed on him by others, that he is doing it because he is forced to, and the consequences of his activity will not be a source of pride to him. Since he feels that external factors determine his fate, the Pawn does not consider carefully his goals in life, nor does he concern himself about what he himself can do to further his cause. Rather he hopes for Lady Luck to smile on him. He is a passive receiver of the 'slings and arrows of outrageous fortune,' always hoping for the one great chance occurrence that will change his life."

A tremendous disappointment is impending in the educational world once again: a behaviouristic, reductionist approach to the use of the Logo language in elementary school classrooms. Many jurisdictions are presently writing scope-and-sequence charts so that appropriate skills are spoon-fed to the students at the appropriate times. The spirit of the Logo language and its accompanying philosophy is getting lost. Logo is becoming another form of CAI - Computer Assisted **Institutionalization**! We are perpetuating the development of Pawns, not encouraging the growth of Origins.

DeCharms (1971, p403) asks,

> How do you teach a person to act in a way that will be successful? Very briefly, you teach him, first, to know his own strengths and weaknesses; second, to choose his goals realistically, taking careful note of his own capabilities and the realities of the situation he is in; third, to determine concrete action that he can take *now* that will help him to reach his goal; and fourth, to consider how he can tell whether he is approaching his goal, that is, whether his action is having the desired effect "

Bereiter and Scardamalia (1983) talk in terms of "helping children to take charge of their own minds". Much of the literature says that "learning tends to be tied to the context in which it was acquired" (p12). The question then becomes: how do learners acquire generalizable knowledge? It is suggested that one should develop a context for **learning about learning** and **thinking about thinking**.

> "You come to understand your own thought processes, to recognize when you are comprehending something and when you are not. Becoming skillful in this context means developing strategies for analyzing, planning, problem-solving, etc. -- strategies that are tied to abstract characteristics rather than to concrete cues." (Bereiter & Scardamalia, 1983 , p 16)

"Nothing great in the world is achieved without passion." said Hegel.

What I find appealing about Bereiter & Scardamalia's approach is that they don't only speak of cognition and metacognition.

> "One of the weaknesses of cognitive process theories in general is that they have trouble accommodating notions of affect and motivation. They thus convey the impression of human beings as dispassionate information processing machines." (p18)

Deci (1980, p18) also says,

> "cognitive theories fail to give proper consideration to the role of emotions in the motivational process."

In my presentation I will detail several practical classroom methods that, not only allow the learner to be self-determining, but provide the child with tools for **thinking about thinking**. Brief descriptions are given below.

AFTERBUGS

This is a technique that can be developed nicely within the culture and context of an active classroom. It essentially excites children about looking for 'mistakes' in their own work.

WATCH ME THINK!

Borrowed from the field of cognitive science, this method involves students working together - one more expert that the other. The *expert* thinks out loud while solving the problem. The *novice* watches and listens to the **process** of the learning and doesn't just view the **product**.

METAPHORIA

I sometimes call this one **Where in the World Are You?** It involves the student in looking for analogous situations to the problem at hand. Metaphors are often used in making a difficult concept understandable and this is useful - but to have the child build her own metaphor or discover her own analogy involves a powerful affective component relating to ownership and personal experience.

OTWID

Oh, That's What I Did! Normally a cognitive science data collection technique, dribble files of keyboard/computer interaction can be used by the student. Why keep this data only for the researcher! It can aid the student in thinking about what they did to produce the end result (desired or not). If only a synchronized think-aloud could be played back too!

GO WITH THE FLOW

(as in Mihaly Csikszentmihayli's descriptions of **play** and **flow**)

One of the complaints I hear often from Logo critics is that children frequently produce incredible patterns, yet don't understand how it was produced. They may have been attempting to produce something else and a 'bug' caused an unexpected pattern. I don't have a problem with these occurrences (as long as it is not the only kind of experience the student is having). Some great scientific and artistic discoveries have arisen out of unexpected bugs and the playfulness exhibited by the creator.

MAKE A BUNCH

Baron (1981, p301) says,

> "telling a person to enumerate more possibilities before evaluating any, to weigh evidence more heavily relative to his prior beliefs, or to set a higher standard for quality of his decisions, ought to have some effect on what he does."

Students should be encouraged to generate many ideas before proceeding.

These ideas can be used effectively with Logo and can be, of course, used at other times too. That is their advantage. They are domain independent! Pre-school children are such incredible learners. Schools should continue to **empower** them - not **deflower** them!

BIBLIOGRAPHY

Baron, J. (1981) Reflective thinking as a goal of education. Intelligence, 5, (pp. 291-309)

Bereiter, C. & Scardamalia, M. (1983) Schooling and the growth of intentional cognition: Helping children take charge of their own minds. In Z. Lamm (Ed.), New Trends in Education. (pp. 73-100). Tel-Aviv: Yachdev United Publishing Company.

DeCharms, R. (1971). From Pawns to Origins: toward self-motivation. In G.S. Lesser (Ed.), Psychology and Educational Practice. (pp. 380-407) Illinois: Scott Foresman.

Deci, E. L. (1980). The Psychology of Self-Determination, D.C. Heath & Co., Lexington, Massachusetts.

Spider Webs and Turtle Walks

Michael Eisenberg
Learning and Epistemology Group, MIT

This talk will describe some preliminary research in modelling the web-building behavior of certain orb-weaving spiders (primarily *Araneus diadematus*) by using Logo turtle programs. The spider's web, built by an animal with poor eyesight and about 30,000 neurons, is an astonishing tour-de-force of engineering. Trying to elucidate the spider's method through a computer program can provide new insights into the web-building process.

A number of interesting side issues may also be addressed through this sort of animal-behavior modelling. For example, if we can develop an adequate algorithmic characterization of the web-building process, we would expect to see spider taxonomy reflected in web algorithms; that is, closely related species should employ similar algorithms. In fact, web-building algorithms may ultimately be used as a technique to corroborate the taxonomic distinctions arrived at by other methods (such as DNA analysis). On a different front, many researchers have studied the effects of drugs like caffeine and LSD on the web-building process; a good algorithmic specification could illuminate these results by showing which particular steps in the orb-weaver's "internal program" are affected by which particular classes of drugs.

Teaching Turtles to Dance
and
Using Turtles to teach Dance

My name is Jody Hemingway and I am presently a Junior at the Agnes Irwin School, a college preparatory school for girls, grades K - 12 in Rosemont, Pennsylvania. I have taken one semester of a Logo computer course, and I am now undertaking a semester of Pascal. Computer is something I enjoy, and I have even taught word processing and Logo to elementary school children at our school for the past three years. In the past year, I have seen computer both as a teacher and as a student.

Christina Reeves, a sophomore at Agnes Irwin, is a ballet dancer. She and I are currently in the process of putting together a project which will unite two seemingly dissimilar concepts, and making them work together as one. With Christina's knowledge of dance and my experience on the computer, we hope to join the two concepts and create a useful tool for teaching the kids. We think we can learn a lot from this too. Here is what we hope to do.

First we will create Logo procedures that simulate several dance moves for the turtle. Leaps and jumps can be programmed by simply hiding the turtle and making it appear in a different spot.

```
TO LEAP :DISTANCE
HT :DISTANCE ST
END
```

A turn, or a faster version, a spin, can be created by repeating a turning action 360 times. For a faster version, we can use fewer steps

```
TO SLOW.TURN                               TO FAST.TURN
REPEAT 360 ]RT 1[                          REPEAT 120 ]RT 3[
END                                        END
```

We may introduce inputs to control the rate of turn and the amount of turn, we are not sure. It depends on the ability of the students. We also plan to make the turtle walk and run, using a repeat command as follows.

```
TO WALK :DISTANCE :SPEED
REPEAT :DISTANCE/ :SPEED ] FD :SPEED[
END
```

We are contemplating more intricate procedures as well. Slides, forward rolls, and hops could all be incorporated into a basic dance vocabulary. The reason for the simplicity of the steps is so that our students can perform them on their own easily. Besides, we can always create more daring moves.

Having developed these basic moves, Christina and I together will present this project to an elementary level, and teach the child to teach the computer how to dance. First, Christina will lead the children through the various dance moves that I have taught the computer to perform. She will demonstrate how these steps can be combined to make more complex maneuvers which in turn are combined to create a dance. It is our hope that after these lessons, they will put together a dance of their very own. While Christina is teaching them the movement, I will be showing them my interpretation of those moves on the computer. I hope to teach them enough Logo so that they can edit procedures and create the more intricate maneuvers from the basic ones. We want them to program their dance onto the computer, then watch and enjoy the turtle´s performance. We will examine the results with them. How are the two dances the same? How are they different? In what ways, is the turtle´s dance better or worse than the actual real life performance. Does the turtle help improve the live performance and vice versa? From there we expect that they will go back and improve their dances in different ways. For example, props like tables and chairs can be incorporated to both dances. We hope we can be ingenious enough that we will be able create the tools they need to enact their ideas. If all goes well, they might even present the dances to the school in an assembly or at our May fair.

Christina and I are just getting started in this project, but we are excited by it´s possibilities. We don´t know exactly how we will do, but would appreciate the opportunity to tell people at the conference. If nothing else, Christina is going to learn something about Logo and I will learn about dance. The children should learn about both and hopefully we will all have fun in the process.

Jody Hemingway & Christina Reeves
c/o Richard Binswanger
Agnes Irwin School
Rosemont, PA 19010
215-525-8400

The Computer as Weather Center

Weather and weather objects are rich with opportunities for exploration and discovery by students of all ages. Logo provides a natural mechanism and environment for organizing, modelling, and exploring weather behavior as microworlds.

High-resolution surface weather maps at national or regional level and up-to-the-minute radar maps for national or regional scales are available. Lists of current observations or historical (climatological) records are available for many stations around the United States and the world.

This information is readily gotten from a number of sources, either free or at modest expense, by telephone to anyone with a properly equipped personal computer.

This graphical information, as well as the lists, may be captured for home and/or classroom use. Once captured the information may be re-viewed at leisure; if desired it may be migrated into a Logo environment or microworld for further exploration.

This presentation will:

- describe and identify several sources of information,
- describe in general terms the hardware and software requirements to access the information, and
- describe capturing such information for local use.

John R. Stremikis
University of Wisconsin Extension
Room 505
432 North Lake St.
Madison, WI 53706

AIBLE

Harry Pinxteren

LOGO CENTRUM NEDERLAND

In this session an AI Based Learning Environment, called AIBLE, will be presented.

Specifications and design principles of AIBLE have been discussed in the context of four EC projects (i.e. projects initiated by the European Commission). An example is the TEMPO project which is part of the ESPRIT II program. TEMPO has recently been started and is meant to stimulate, among other things, the use of AI software both in schools and at home. So, AIBLE could stand as an example of a potentially important line of thinking in this remote part of the world. Some aspects of this thinking might be significant for an international LOGO exchange and will therefore be discussed in more detail.

AIBLE has been designed as an integrated, machine-independent, *low cost* , state of the art system. It does not force upon the user one single style of programming or problem solving. It allows for the combination of a conventional, procedural style (e.g. Turtle Graphics, LOGO), a functional style (LISP) an object oriented (SMALLTALK-based) and a declarative (PROLOG) approach. In addition, - given its central interface - , AIBLE offers options for the integration of various applications which, in most cases come as *extensions* of the system. So, it covers a whole range of general purpose and dedicated software and meets two important criteria set by the TEMPO project. It is both a 'vertical' and a 'horizontal' system, cutting across various domains and levels of expertise.

Some practical implications of these features will be discussed. As an example the *top level editor* of AIBLE will be presented. Top level means, among other things, that the editor is the main interface of the user with the system: communication is very direct and not interrupted by switching between 'direct' and 'indirect' mode. In addition interaction is supported by a *visual syntax* (more advanced than BYSO LISP's visualization and pretty close to BOXER, LOGO's alledged successor).

Visual syntax means that the structure and working of LOGO programs are displayed as (coloured) windows, called PLANES. In short, LOGO programs, are formed as PLANES following one simple syntax rule (there is no exception to this rule). They can be manipulated directly, very much as sheets of paper, by using a pointing device. So, for example, PLANES (i.e. parts of LOGO programs) can be inserted, deleted, copied and executed. PLANES are also used to support *automatic error detection* (also, semantic) by showing *where exactly* what kind of error has occurred, thus focussing the attention of the user to the problem at hand. This feature amounts to a built in trace and debug facility. In combination with other features of this editor, which will be illustrated, and an object oriented I/O, AIBLE can be seen as an intermediate step between BOXER and common LOGO versions.

Some concrete, practical applications of a PLANEd LOGO will be given, e.g.: in the field of arithmetics and language.

For further information:

H. Pinxteren
LCN
PO BOX 1408
6501 BK Nijmegen, The Netherlands

Visualizing LOGO syntax: a structure editor

Venant Haars
LOGO Centrum Nederland

The preoperational child has been described (e.g. Piaget, 1953) as finding it difficult to think about part-whole relationships. For example, it has been shown that children and even adults often have difficulty in solving class inclusion problems (for a review, see Winer, 1980). There is general recognition that class inclusion is a prerequisite to logical reasoning (Braine & Roumain, 1983; Haars & Mason, 1986; Kuhn, 1977): in order to successfully solve logical problems, the student must be able to recognize the superordinate and subordinate relationships involved. In other words: he or she must be able to grasp the *structure* of the problem.

A student who is busy writing LOGO procedures is essentially engaged in logical problem solving. I believe that when children are having trouble designing and debugging their procedures this is not necessarily so because of a lack of programming skills, but because they lack sufficient understanding of the *structure* of the problem at hand.

One of the ideosyncrasies of current LOGO implementations is that the structure of expressions is usually hidden. For example, nesting of REPEAT-statements is denoted by brackets; this way many things are occurring together on a single line of code. As a consequence, children often find their own procedures unreadable.

The Dutch LOGO Center has developed a toplevel editor that provides direct access to structures according to the principle "what you see is what you have". It uses coloured planes to enter and manipulate LOGO expressions. That is, arguments and the (sub)expression to which they belong are displayed as a single coloured plane. By moving the cursor through a procedure, all subexpressions and their arguments are subsequently displayed as planes. Thus, planes visualize the hierarchical structure of LOGO expressions. Only one plane is actually shown at a time (in reverse video), while the rest of the text is displayed the usual way.

With a single key-stroke the current plane can be evaluated, after it has been selected by pointing at it with the cursor. This way subexpressions can be evaluated separately, and this feature appears to be especially useful during the debugging stage. Planes can be copied, deleted, appended and inserted as if they were sheets of paper. Also, planes may be sent to a printer, or saved on disk or tape.

In this presentation the use of this new editor will be demonstrated and some of its educational implications will be discussed.

References:

Braine, M.D.S. & Rumain, B. Logical reasoning. In: P.H. Mussen (ed.) *Handbook of Child Psychology (4th ed.), Vol. iii: Cognitive Development.* New York: John Wiley, 1983.

Haars, V. & Mason, E.J. Children's understanding of class inclusion and their ability to reason with implication. *International Journal of Behavioural*

Development, in press.

Kuhn, D. Conditional reasoning in children. *Developmental Psychology*, 1977, 13, 342-353.

Piaget, J. *Logic and Psychology*. London: Manchester University Press, 1953.

Winer, G.A. Class inclusion reasoning in children: a review of the empirical literature. *Child Development*, 1980, 51, 309-328.

For further information, please contact:

Venant Haars
LOGO Centrum Nederland
Postbus 1408
6501 BK Nijmegen
the Netherlands

Developing Educational Software in Logo

Introduction

The claims that Logo helps students to learn mathematics, develop problem solving skills, learn programming and develop confidence in their own abilities are well known. Most of these claims are made in the context of Turtle Geometry - a very mathematical computational learning environment.

If Logo is to be relevant to the needs of a wide range of students it must be possible to use Logo as a development language to produce computational environments which can be used to support learning in different parts of the curriculum.

This paper outlines an attempt to produce Logo based environments in science. The work is in progress at the Educational Computing Unit, Kings College, University of London.

Logo as a software development language

Educational software is traditionally written in such a way that the development language is hidden from the learner. Given that the design criteria of most languages are not educational in nature this is probably a good idea. The case with Logo is exactly opposite - the design was motivated by educational rather than technical considerations.

If educational software can be written in Logo in an open style i.e. in a way which enables the language to be used within the software itself, then we will have produced learning environments which epitomise the power and philosophy of Logo in specific contexts.

We see two components to the development of such educational software:

a) The creation of a 'first class' computational object[1] which will serve a similar function to a turtle. This object could be a novel creation or an extension of the traditional turtle.

b) The development of a suite of procedures, written in Logo which provide a starting point for the learner.

These ideas have been used to develop a computational environment concerned with the concept of a field in science.

[1] For a discussion of the term 'first class object' see 'Thinking about (TLC) Logo, J Allen et al. For the moment the significance of the term is that we would wish communication with any new creature we introduce to Logo to be as consistent and understandable as the usual chat - such as forward 10 , penup , etc.

A field turtle

In attempting to develop a new first class object we have taken a 'conventional' turtle and added to the list of primitives which define the turtle's state . Ordinarily these are things like **heading**, **penstate** and **position**. To this list we have added a **window** primitive through which the turtle can view its environment. We have put this turtle in a field and its movement will be determined by its interaction with the field as viewed through its window. Figure 1 shows a turtle with a rectangular window moving in a field represented by a rectangular grid (or list of lists) Each grid position is occupied by a number representing the field strength.

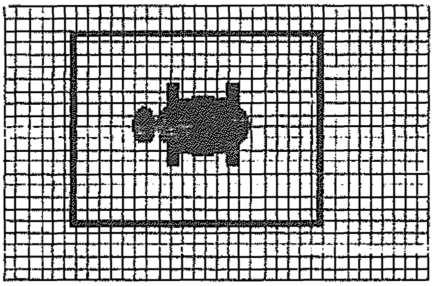
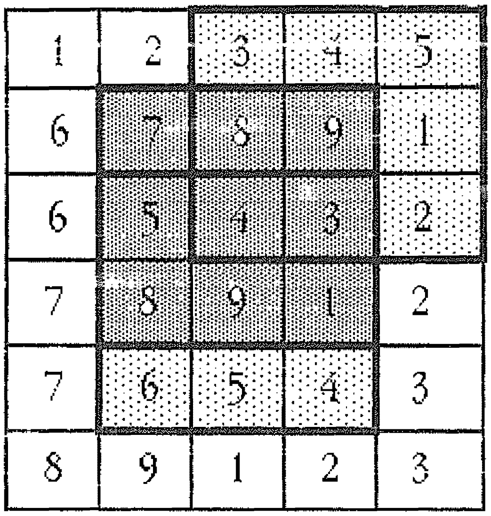


Figure 1 A field turtle with a rectangular window on the field.

The window primitive can take inputs to define its size and shape. The way that the turtle interacts with the field will be governed a rule which is implemented as a procedure. Different physical situations can be explored by changing the procedure.

An example of a rule would be to require the turtle to move 8 adjacent grid positions and for each position to total the 'size' of the field seen through the window. The rule could then require the turtle to move to the position corresponding to the lowest total. Figures 2 and 3 show this process in action.



1	2	3	4	5
6	7	8	9	1
6	5	4	3	2
7	8	9	1	2
7	6	5	4	3
8	9	1	2	3

Figure 2 Some field evaluations.

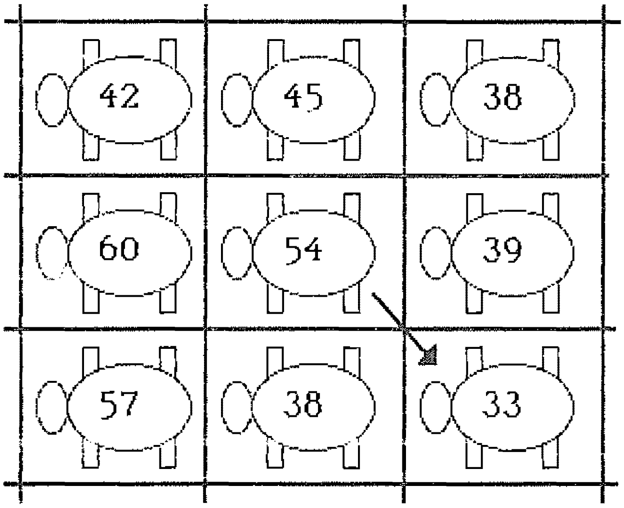


Figure 3 The turtle moving.

We can provide procedures which describe how two turtles will interact in a field. The procedures will invoke rules which are valid in the area where the turtles' windows overlap. This is shown in Figure 4.

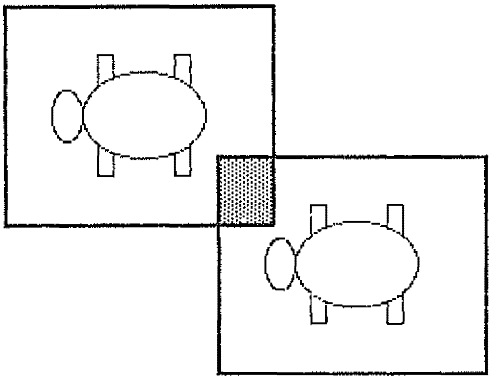


Figure 4 Interacting turtles.

David Squires
Royston Sellman
Educational Computing Unit
Centre for Educational Studies
King's College London
University of London
552 King's Road
London, England SW10 OUA

A DIFFERENT ANGLE OF VISION:

WHAT LITERACY RESEARCH HAS TO OFFER LOGO USERS

I am not, and would never claim to be, an authority on Logo. In fact, most of my research has studied the impact of other forms of computer use on literacy. However, I think many of my findings have much to suggest to Logo users.

As educators, we teach students exemplars for use in problem-solving. Proficiency with these exemplars constitutes the core of literacy (not mastery of a collection of skills). In The Structure of Scientific Revolution, Thomas Kuhn described an exemplar as a shared problem-solving example - "a time-tested and group-licensed way of seeing." When students study a particular discipline like science, they learn exemplars deemed useful by that disciplinary community through their application to a variety of problems.

Outside the boundaries of a formal discipline, however, educators model more generally applicable exemplars. These exemplars derive from strategies necessary for coping with the symbol systems we use to encode and decode what our culture has come to know. Until recently, print has been the predominant symbol system, and, as a consequence, traditional problem-

Prof. Priscilla Norton
Dept. of Curriculum and Instruction
University of New Mexico
Albuquerque, NM 87131

solving exemplars taught to students have focused on logical structure, on linear, sequential reasoning, and on pre-planning and planning. Inherent in these problem-solving exemplars are the search for a single solution to which all can agree and the search for cause and effect relationships and propositional statements. In short, these exemplars emphasize the kinds of individual puzzle-solving capacities generally associated with the current problem-solving research paradigm.

However, when using this research paradigm, Logo and other computer applications often fail to live up to their claims, and this failure challenges the possibilities of the revolution first proposed by Logo users. To resolve this problem, we need to do two things. First, we must explore the implications of Logo use in particular and computer use in general within the frame of the larger, emerging demands of the social context. Reports from the business community, for instance, are calling upon educators to teach not the content associated with a vocation but processes such as efffective communication strategies, cooperation strategies, problem-solving, creating, and learning to learn. Learning with computers as opposed to books suggests learning the very processes demanded by the larger social context. Two recent research projects using a variety of computer software support this claim, although little Logo research is set in this context. Yet, it is in exploring these demands that at least part of the radical changes suggested by Logo will be realized.

The second trap set by the current problem-solving research paradigm centers on our unwillingness or inability to distinguish between a description of process (a book) and process itself (a computer). When I am involved with process, I ask what-if questions. When I describe the outcomes of that process, I report if-then or propositional results. When I am involved with process, I explore probabilities and search for variables and rules of interaction. When I describe that process, I present single solutions and cause and effect relationships. When I do process, I rely on intuition and creative insight. When I describe process, I speak of pre-planning, planning, and breaking the problem into parts. If we are to investigate problem-solving in Logo, a learning environment like simulations and word processing where descriptions and processes come together, we can not look only for reflections of traditional problem-solving exemplars. Rather, we must look beyond exemplars which describe process to exemplars associated with doing process. Some of these possible exemplars will be explored in the presentation.

Documenting Logo growth: A developing methodology

Logo 86

Marita R. Hopmann
Child Development Center
Rhode Island Hospital

J. Michael Hopkins
St. Paul Public Schools
Macalester College

Nelson H. Soken
Macalester College

Progress in understanding children and their development in the context of their work with Logo is dependent upon the collection of systematic information. The potential beneficiaries of effective methods of gathering information about children's work with Logo are many: (a) teachers, who wish to corroborate their informal impressions, (b) administrators, who need documentation to demonstrate the activities of the students and teachers , (c) parents, who want to know more about what their children are doing while at school, (d) researchers, who need systematically-gathered information to address questions about children and their development, and, (e) children, who will benefit by increased attention to the process of their learning.

At this time an increasing number of students are using Logo in our schools, with the expectation that Logo experience will be beneficial in a variety of domains, including problem-solving, commùnications skills, and self-esteem. Nevertheless, little evidence of change in students has been presented to support these claims. One reason for the paucity of data has been the difficulties associated with effectively documenting children's Logo work. This paper describes a method of collecting data, and a process for making sense of the data, using as a case study an examination of the topic of errors and feedback.

Data Collection Methods

Our new method of acquiring systematic information about children's work with Logo takes advantage of the availability of video tape recorders in most of our schools. We used a VCR to record the signal which the computer sends to the monitor, attaching video cables from the computer to the VCR and from the VCR to the monitor; in this way we recorded everything that the children typed without interfering with their work. By attaching a microphone to the audio input of the VCR, we were also able to record the conversations that occurred between the children at the computer.

Our second step consisted of transcribing from the video tapes everything that had been typed, following a temporal sequencing. Finally, we applied a coding system to the transcriptions, allowing us to summarize the students' work. The striking power of this technique derives from the fact that all of the students' work is available for analysis. With limited resources, a teacher could use this relatively simple method to document student growth over time. In addition, researchers wishing to test specific hypotheses have access to an abundance of raw data. One can quickly examine the transcriptions to look for evidence of specific skills or attributes, such as the use of procedures and subprocedures, finding the behavior of interest in the context of the rest of the work during that session. Occasional tapes of students' work can be used directly as exemplars and may be useful in illustrating change over time.

A Case Study: Children's Use of Errors & Feedback

The Logo activity of twenty third grade students, working in dyads, was recorded on two occasions, about six weeks apart. A trained observer was present to set up the recording instruments and to observe and record aspects of the social interaction which could not be retrieved from the tapes. The transcriptions from these sessions were analyzed to gain information about the children's use of erasing, and their responses to error messages. We had noted that the children sometimes erased sequences which would have been "ungrammatical", but before an error message was produced; this indicated to us the use of the visual medium as an aid to see the discrepancy between one's intention and one's immediate behavior. On other occasions, the children erased perfectly "acceptable" sequences, suggesting a change of mind, either in response to seeing their commands represented visually in front of them, a simple second reflection, or a planned variation.

We wondered how often the children used the information in the error messages to "correct" their work. It is widely believed that a great benefit of using a computer is the deligence and limitless patience of the machine in pointing out errors. What do children do with the information they receive about errors ? How do they respond to the feedback the machine offers?

From preliminary analyses of the transcripts, we found that these children were very actively engaged, producing between 3 and 4 lines of commands per minute. Their utterances averaged two-and-a-half commands, showing that the children were operating in units considerably larger than single commands.

A large number of the children's utterances had erasures in them. The erasures were evenly divided between those made in otherwise "grammatical" utterances, and those made to correct errors. Half of the changes made in the grammatical sequences were performed on utterances which had been duplicated via Control Y for the express purpose, it would seem, of using a frame and changing a critical part. These often occurred in sequences of many repetitions and changes.

While some measures of the children's level of sophistication would suggest a rather primitive stage of functioning (e.g., infrequent use of procedure-writing in the editor, limited use of repeat commands, and infrequent evidence of clearly planned object-designs), the use of Control Y with erasure by six of the ten pairs reminds us of the various dimensions of problem-solving which work with Logo can entail. This operation requires fairly sophisticated use of the system, and self-conscious planning in the utterance generation.

Error messages were produced following over 10% of the children's utterances. We distinguished between the first and second responses following an error message, noting both when the children attempted to correct the cause of the error message, and also when the error messages were followed by multiple carriage returns, erasing the error message. Perhaps adults' views of the gentle qualities of Logo in pointing out one's errors in a nurturant manner are not shared by children.

In summary, we have established some baseline information about important components of problem-solving work: overall rates of productivity, lengths of child-determined units, child-initiated changes, rates of error messages, and responses to error messages. By presenting these pieces of the puzzle in terms of behavior from transcripts, others will be able to reflect on common and unusual aspects of the activities of children with whom they are working, and in this way information can more easily be shared. This process of demystifying grand concepts, by proposing ways of operationally defining them, empowers members of the Logo community by making it more feasible to share observations, and should help teachers communicate with their multiple partners in education: children, parents, and administrators.

THE POWER IN LOGO DRIBBLE FILES

John Olive and Cheryl A. Lankenau

The Atlanta-Emory Logo Project
Emory University

INTRODUCTION

The inclusion of a Dribble capability with LCSI's Apple Logo II opens up avenues for data collection of students' interactions with Logo which have been available previously only on main-frame versions of the language. This rich data base can be used for research, curriculum, instruction and evaluation purposes.

Executing the Dribble process in Apple Logo II causes all text that is printed on the screen to be dumped to a file on disk. Thus, every keystroke the student makes while working with Logo is captured in this file. The data are stored as text files which make them accessible for printing. Used in this way, hard copies of each student's work session are available for analysis. Additionally, it is possible to playback the actual work session on a computer, using a set of special Logo procedures.

The Atlanta-Emory Logo Project is a three year investigation into the use of Logo to help ninth grade students aquire understanding of geometric relationships. Students' Dribble files are a major source of data for our study. The Dribble process in Logo II does not capture the edit screen; consequently we have redefined the edit functions in Logo II so that changes made in the editor will be displayed on the text screen and thus become part of the Dribble record. We have also developed a set of Logo procedures which make it possible to playback the Dribble files. In the playback mode, the student's commands are processed and displayed on the screen. The student's procedures are defined and redefined after every record of an edit. A complete, dynamic, visual recording of every step a student takes while working on a Logo task is thus obtained. This visual recreation, together with the printed record, enables us to analyze the learning process in great detail.

We are, however, only just beginning to explore the power inherent in the Dribble process. This power is available to all of those directly involved in the learning process: students, parents, teachers and researchers. The focus of our session will be on the presentation and discussion of specific goals and objectives for using Dribble files. We shall share activities for the use of Dribble files by students, teachers and researchers and anticipate a brainstorming session to generate deeper exploration of these issues for all target groups.

STUDENT USE

The playback of Dribble files offers students a concrete opportunity for reflection and evaluation of their own and other students' learning processes. Such metacognitive skills are important in the development of higher-order cognitive processes, such as:

a. Identifying (after the fact) effective and non-effective approaches to a problem or task.

b. Anticipating and interpreting reasons for difficulty.

c. Exploring and benefitting from errors and feedback -- understanding the value of failure.

d. Learning to evaluate work according to specified criteria.

e. Learning to establish evaluative criteria.

f. Looking for alternatives to one specific problem (divergent thinking).

g. Elaborating on one's own and other's solutions.

Student use of Dribble files could also help in the development of affective processes. For example:

a. Establishing curiosity about one's own process of learning.

b. Generating self-confidence through reflection on and appreciation for one's own efforts as reviewed in the Dribble files.

c. Valuing growth in the learning process.

TEACHER USE

The use of Dribble files by teachers could help them develop insight into their students' learning processes and the effectiveness of their instruction. It could provide information on individual student's progress and can enable teachers to become researchers within their own classrooms. For example, teachers could use Dribble files:

- as an evaluation tool for specific activities
- as an evaluation tool for teaching effectiveness (were instructional interventions appropriate for individual students?)
- as an aide to understanding individual student's programming style, problem-solving strategies, and cognitive styles and strategies
- as a record of individual progress.

RESEARCHER USE

By capturing the dynamics of learning processes, as well as their outcomes, and through the documentation of individual differences in learning procesees, we can validate and refine current theories and develop new theories. Researchers need to develop criteria for the analysis and evaluation of Dribble files (Olive, 1985) and to develop procedures for effecient use of the vast amount of data generated by the Dribble process.

LIMITATIONS

Although we recognize the potential power in the use of Dribble files, several limitations have emerged through our experiences in the Atlanta-Emory Logo Project:

a. No indication of time-span (other than a class period).

b. No record of verbal interactions among students or between student and teacher.

c. Difficulty in determining which data are the results of instruction or direct intervention -- the need for lesson plans and observation notes to interface with Dribble files.

d. Dependence on student to initiate and terminate the Dribble process.

e. Technical difficulties if the Dribble process is not terminated.

f. Technical limitations of current hardware and software (frequent disk drive problems when using Logo II).

We look forward to discussing these limitations in the hope of generating technical or procedural solutions.

Examples of Dribble files from the Atlanta-Emory Logo Project will be shared to demonstrate the value of the computer playback facility.

References

Olive, J., 1985. A study of the application of a qualitative taxonomic synthesis to the analysis of geometric reasoning in a computer environment. Doctoral dissertation. University Microfilms International, Pub. No. 85-16,579. Ann Arbor, MI.

Author's Notes

The Atlanta-Emory Logo Project is supported by grants from the National Science Foundation (grant # MDR-8470287) and Apple Education Affairs (grant # 3658).

The set of Logo procedures for playing back the Dribble files and the redefined edit functions for Logo II were developed by Dr. Olive specifically for the Atlanta-Emory Logo Project. Copies of the procedures will be available at the session.

For more information please write to:

Dr. John Olive, Project Director
Cheryl A. Lankenau, Project Evaluator
The Atlanta-Emory Logo Project
Division of Educational Studies
Emory University
Atlanta, GA 30322

Iconic Programming

A strategy and metalanguage that helps one learn the "hard parts" of Logo

Often our goal in using programming is to offer students the opportunity to manipulate, in concrete and dynamic ways, ideas that may otherwise be experienced as static and abstract. We must then take care that the programming does not become its own curriculum, that it does not draw so much attention to itself that it obscures the subject it is intended to teach. Logo may, in itself, have "no ceiling," but people experience a ceiling when they cannot learn how to get Logo to do more for them.

The ubiquity of Turtle Graphics and near non-existance of the use of list-processing in Logo programming with children is a reflection of this reality. Turtles are supposed to be "easy;" lists and output are supposed to be "hard." But, what is "easy" and what is "hard" depends very much on the programming models we use in our teaching, and the metaphors with which we explain them, especially at the beginning.

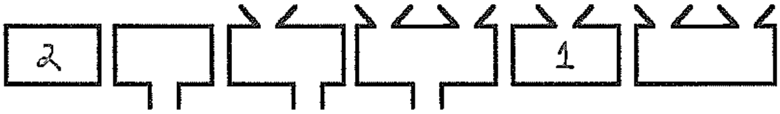
Deciding on an appropriate programming model to begin with is essential. Compare these two programs for generating sentences:

```
TO PROGRAM1                                        TO PROGRAM2
 PR (SE PICK :PERSON PICK :ACTION PICK :PERSON)     OP (SE PERSON ACTION PERSON)
END                                                END

TO SETUP                                           TO PERSON
 MAKE "PERSON [Dale Dana Sandy Chris Lee]           OP PICK [Dale Dana Sandy Chris Lee]
 MAKE "ACTION [loves [sings to] hates kisses]      END
END
                                                   TO ACTION
                                                    OP PICK [loves [sings to] hates kisses]
                                                   END

?SETUP
?PROGRAM1                                          ?PR PROGRAM2
Dale sings to Chris                                Dale sings to Chris
?PROGRAM1                                          ?PR PROGRAM2
Sandy loves Dana                                   Sandy loves Dana
```

Which is more easily learned? Put that way, the question is purely empirical, but there are analytic considerations as well. Which is "simpler" depends upon your metric. PROGRAM2 is longer and requires PR in its invocation. But it is also a more consistent programming model, it requires less knowledge of Logo (OP instead of dots, quotes, and MAKE), it does not require the SETUP step before use, and it more readily generalizes to advanced programs.

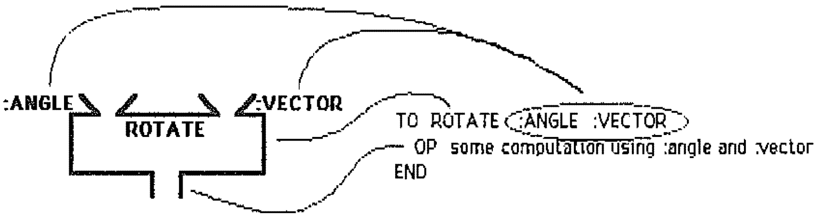
Not quite independent of one's choice of model is the decision about what metaphor to use. Metaphor is often established just by the language we use, and without our consciously thinking about it. When we say "and then it goes back up to the top and does it all over again," we are teaching students to think (incorrectly) that recursion is a kind of loop. The language we use also determines whether a student thinks of MAKE as putting a value in a named box or as taping a name on a value. In this case, the issue is not so much which image is correct, but which is more serviceable or learnable. One of those questions is empirical, the other essentially theoretical. In any case, if we let the metaphor drift haphazardly with the words we happen to choose on a particular day, we risk creating an unnecessarily complex image of Logo programming.

An iconic "programming language" with Logo syntax helps elementary and high-school students—rank beginners—to build data-bases, manipulate points in Cartesian space, model word-problems in mathematics, experiment with probability, and even model their own language, all using procedures that output. In fact, though serviceable for all Logo programming, this iconic representation is particularly well suited to the "hard" parts of Logo. With the icons comes a new vocabulary, a meta-language with which to talk about programming. Logo procedures—primitive procedures as well as ones defined by the user—are represented as machines, with or without hoppers for input, and with or without an output spout, as appropriate. I propose the names solo, source, processor, and destination for the resulting four-way taxonomy. The distinction between operations and commands (procedures that do and do not output, respectively) is traditional, but, especially for the beginner, the distinction between procedures that do and do not require inputs is equally important and deserves its own terminology. Larry Davidson has suggested that they be called, respectively, consumers and self-starters.

	SELF-STARTERS			CONSUMERS		
type	SOLO	SOURCE	PROCESSOR		DESTINATION	
icon						
primitive examples	CS POTS CATALOG SQ TRI	HEADING POS XCOR	FIRST BUTFIRST SQRT EMPTYP	SUM LIST SE PRODUCT ITEM MEMBERP	PRINT SETPOS FD SAVE PO EDIT OP	REPEAT MAKE

Logo instructions consist either of a solo (e.g., CS) or of a destination and its inputs (e.g., REPEAT 5 [FD 100 LT 144]). Inputs are expressions that may be literal objects (words or lists, as in the example just given), named objects (e.g., :SHAPE), source procedures, or processors with all of their inputs (e.g., SE "FD SE SE WORD SUM -41 51 0 "LT 144).

My own inclination is to teach Logo procedures, structures, and syntax in the context of some motivated project, putting cart and horse in what I believe to be their proper order. I am not inclined to make up arbitrary examples of FIRST, LAST, and LIST just to explicate their behavior or to bludgeon my students with the vocabulary lesson I just gave you. However, when I do teach new procedures, I teach them in their iconic notation. When students design their own procedures, I encourage them to program first with the icons, and I teach them explicitly how to translate from that iconic notation to the conventional Logo text representation and vice versa. In particular, the icon fully specifies the skeleton of the procedure.



```
TO ROTATE :ANGLE :VECTOR
OP some computation using :angle and :vector
END
```

What one does using this icon language depends, of course, on what one cares to teach (or learn). Just as the choice of model and metaphor matter, so do some variables in the choice of learning projects. Clear examples of iconic programming and a rationale for choosing among projects in language and mathematics can be found in my papers on those subjects (see "Exploring Language" and "Building Mathematical Systems").

The Bead Necklace:
a Physical Model for Words and Lists in LOGO

Lorne H. Bouchard
Dép. de math et d'info

Louisette Emirkanian
Dép. de linguistique

Université du Québec à Montréal
C. P. 8888, Suc "A"
Montréal, QC H4A 2S3

Abstract

We present the Bead Necklace, a physical model for words and lists in LOGO and show how it can help us plan the steps required to construct and modify complex lists.

Introduction

As heavy users of LOGO as a programming language we believe that there should only be one LOGO; this implies, on the one hand, "no threshold" for the beginner (Turtle Geometry) and, on the other hand, "no ceiling" for the experienced user ("Sugared" LISP). Thus, there is no "growing out of LOGO" as often happens with other programming languages. As teachers of linguistics students, we know that there is a very difficult gap between turtle geometry and word and list processing. Turtle geometry is based on a physical experience - i.e., crawling on the floor - which everyone has experienced before his first birthday. The Bead Necklace, the analog of the floor turtle, is a physical model for words and lists in LOGO which the student can manipulate: this provides a concrete experience of words and lists. A LOGO program can display on the computer screen the bead necklace corresponding to any word or list.

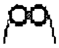
The Bead Necklace

A physical bead necklace is constructed out of wood beads and wire. Small light-colored beads are used to mark the beginning and end of words. Large light-colored beads are used to represent the letters of a word, one bead per letter on which the letter is inscribed. Small dark-colored beads are used to mark the beginning and end of sentences or lists.

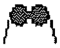
The beads are strung together with wire to form a necklace. There is *at least one and at most two wires* which pass through a given bead of a necklace. After stringing a necklace, the ends of the wires are neatly cut and bent back. When taking the necklace apart, we *must* always remove the wire which is the only one through some bead.

There is a bead necklace for every word and list in LOGO.

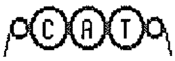
The empty word " is represented by two small light-colored beads held together by a piece of wire which is bent at both end. This is pictured as follows:



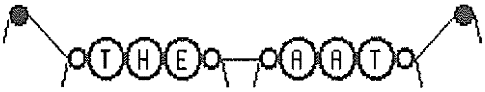
The empty sentence or list [] is represented by two small dark-colored beads held together by a piece of wire which is bent at both ends. This is picured as follows:



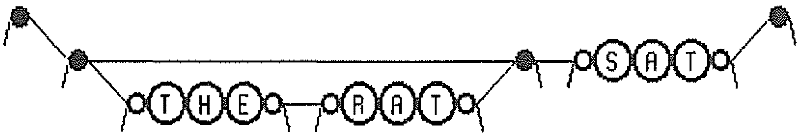
The word "CAT is represented by a necklace of the three large light-colored beads - identified C, A and T respectively - which is terminated by a small light-colored bead at both ends. The wire which holds the necklace together is bent at both ends. This is pictured as follows:



The necklace for a sentence is constructed in two steps. First , we build the necklaces for the words of the sentence; these necklaces are then strung together to build the necklace for the sentence. For example, the sentence [THE RAT] is represented by stringing together on a wire the necklaces for the words "THE and "RAT; this new necklace is terminated by a small dark-colored bead at both ends and the wire which holds this necklace together is bent at both ends. This is pictured as follows:



In a similar fashion, the necklace for a list is constructed by first building the necklaces for its sub-lists and words: theses necklaces are then strung together. For example, the list [[THE RAT] SAT] is represented by stringing together on a wire the necklaces for the sentence [THE RAT] and the word "SAT; this new necklace is terminated by a small dark-colored bead at both ends and the wire which holds this necklace together is bent at both ends. This is pictured as follows:



The procedure we have just described constructs a bead necklace from the bottom up. You can see that at least one and at most two wires thread each bead of a necklace. If we wish to take the necklace apart, we must start at the left or right end of the necklace where only one wire threads a small bead, we call this the top level of the necklace. The procedure for taking the necklace apart is thus a top-down procedure.

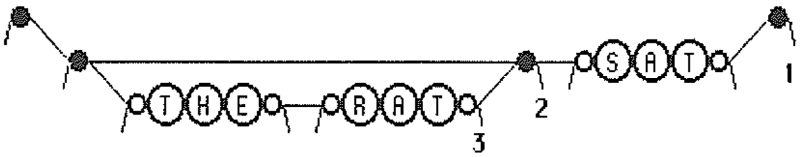
Constructing complex lists

We show how the bead necklace model can help us plan the construction of complex list structures. As an illustration, consider the task of constructing the list **[[THE CAT] SAT]** from an existing list **[[THE RAT] SAT]**. This task divides neatly into three sub-tasks: dismantling the **OLD** list, modifying a part of it and reconstructing the **NEW** list from these parts.

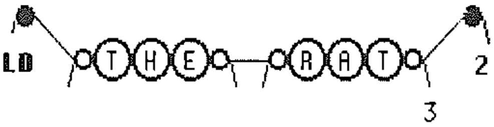
We step through the Logo instructions for achieving the task, using pictures of bead necklaces to draw the result at each step.

The first sub-task takes the **OLD** list apart. This is achieved by manipulating the wires identified **1** and **2**.

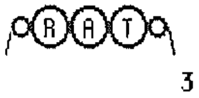
```
MAKE "OLD [[THE RAT] SAT]
```



```
MAKE "STEP1 FIRST :OLD
```

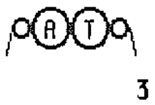


```
MAKE "STEP2 LAST :STEP1
```

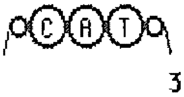


We can now change **"CAT** to **"RAT**: this involves manipulating wire **3**. This wire is relabelled **3'** to indicate that the necklace has changed at this level.

```
MAKE "STEP3 BF :STEP2
```

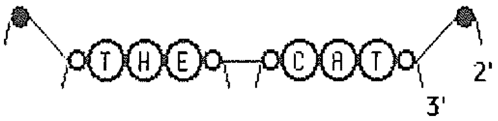


```
MAKE "STEP4 WORD "C :STEP3
```

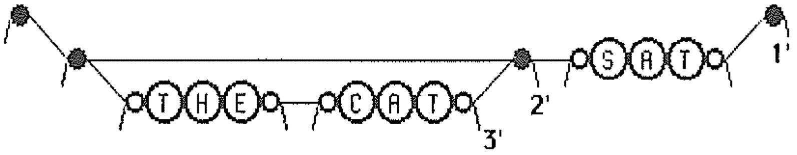


We can now assemble the necklace for the **NEW** list; this is done by reassembling the existing parts. This involves manipulating the wire labelled **2'** and **1'**.

```
MAKE "STEP5 SE FIRST :STEP2 :STEP4
```



```
MAKE "NEW LIST :STEP5 LAST :OLD
```



In order to change the **OLD** list we had to take it apart and rebuild from its parts the **NEW** list. The bead necklace model makes it easy to understand why this must be so and how to go about doing it.

Note that all the pictures of bead necklaces were produced by a MacLogo procedure called **DRAW.NECKLACE** (which is implemented using Turtle Graphics). These pictures help us visualize the abstract objects which are words and lists.

We believe there is a strong analogy between the Bead Necklace and Turtle Geometry:

Level	**Turtle Geometry**	**Bead Necklace**
abstract	turtle geometry	words and lists
visual	screen turtle	screen necklace
concrete	floor turtle	bead necklace

Conclusion

We have presented very rapidly the Bead Necklace model for words and lists in Logo and shown how it can help us plan the construction of complex list structures. If this model can help some Logo users work out their sticky problems involving words and lists, we will consider we have succeeded.

Jeff Sandys Logo 86

DIAGRAMING INSTRUCTIONS

The syntax of a Logo instruction is like an imperative sentence. The subject, the turtle or computer, is understood. Commands are the verbs, and operations are the noun phrases, the objects of the verb.

English sentences can be diagramed as an aid to understanding syntax. Logo instructions can also be diagramed. The simple technique, demonstrated at this session, will find errors in long instructions. It also helps students write longer, more powerful instructions.

Diagraming reveals the importants of operations. It also offers a naming convention for user defined procedures. Students will write more operations to create more powerful programs. They will also write procedures that are easier to understand. A handout will be available at the session.

Project Headlight: The First Year

A team presentation by staff members of the Hennigan School
and the MIT Learning and Epistemology Group

Many research results confirm the direct experience of very many teachers that an hour or two a week with Logo can provide a valuable enrichment of school learning. Project Headlight is a step towards a more thorough integration of the computer, and Logo, into the life of a school. A "school within a school" consisting of 220 city students (grades 1 through 5) and 13 experienced public school teachers has access to over a hundred computers. During a preliminary period a principal goal was to develop computer proficiency by giving the students between 45 and 90 minutes of access to Logo and word processing per day. As proficiency developed, the emphasis has shifted towards gradual integration of the computer work into the teaching of standard school subjects. Methods for doing this are forged in the course of day-to-day collaboration between the teachers and members of the MIT group. The MIT group is responsible also for an exceptionally extensive observation and evaluation study.

The presentation will cover the following topics:

- Descriptions of the disposition of computers (mostly PCjrs) in classrooms and open areas and of the computer system used, including the use of networking and electronic mail controlled from within Logo.
- Reports on the process of learning Logo under these conditions.
- Descriptions of a number of projects in language, science and math which make essential use of Logo programming.
- Brief descriptions of some special projects which will be the subject of independent papers at Logo86 (the Ocko and Resnick presentation on Logo/Lego; Upitis on a Music classroom in the Logo spirit).
- Overview of methodology of the observation/evaluation component and presentation of some preliminary results (some of these results will also be developed more fully in other papers at Logo86, e.g., Harel's paper on Kids as software designers).
- The students are fairly evenly distributed between those classified as "advanced," "regular," and "special needs." The presentation will provide samples of work from these three categories. Special emphasis will be given to some case studies of special needs students who responded particularly well to the learning environment. Hypotheses discussed will relate to the way Logo work can develop self-image and to its value as a medium for students with strong abilities.

The observational work gives particular attention to the recognition of "styles" of work. Some cases will be described that confirm and deepen prevalent concepts but also point to the need for more subtle elaboration of thinking in this domain.

Gender issues are studied and discussed in close relation to the question of styles in general. Findings are very different from those that are reported elsewhere. Discussion will try to sort out how much of this is due to differences in ease of access (there is much less competition for computers than in places where machines are a scarce resource), to differences in the form of the computer culture developed at this school, and to age and social background.

More than a third of the students are Spanish-speaking and one of the goals of the project is to use this circumstance to dig more deeply into how cultural issues enter into the picture. Results in this domain are still quite unclear but some observations will be opened for discussion.

MICROWORLDS AND CREATIVITY IN THE ELEMENTARY SCHOOL CURRICULUM

This paper examines ways in which microworlds can be used as an exploratory tool within the traditional elementary school curriculum. Its starting point is a series of microworlds which were developed as a part of an introductory and an advanced Logo classes during this year.

These microworlds were developed by both practicing teachers and college students. Some of them were then intergrated into a Logo curriculum for grades K - 5 which the presenter has developed for Knox county schools. This curriculum aims to preserve the exploratory and discovery concepts of Logo while at the same time providing sufficient structure to allow diffident, insecure or non-computer oriented teachers to begin to use Logo in their classrooms.

The selected microworlds were then used by their designers or other teachers in the classroom. The presentation will include videotape of this implementation as well as our analysis of the factors involved, the materials developed by students as an outgrowth of these microworlds will be displayed. This discussion will include both Logo and non-Logo materials.

The microworlds which were developed and implemented include the following:

The Straight Line Letters of the Alphabet

This is a multi-tactile microworld for Kindergarten students. This microworld allows students to explore, through the use of robotic toys, drawing tablets, such as the Koala pad, Logo, and other sensory materials, the form and nature of the letters which they are learning in the Kindergarten classroom. It encourages the creative use of the letters as well as providing for their correct formation.

Let's Become String Artists

This microworld allows students to discover and use principles of aestheometry in the 4th grade classroom. Opportunity was given to students to use Logo as a design tool and then to reproduce their design as string art pictures. In this microworld, students were developing an understanding of line and angles and the ways in which geometric shapes are formed.

Now I Can Read

Often list processing is described as the "hard" part of Logo. In this microworld file designers sought to develop tools that would allow first graders to begin to use their newly acquired reading skills in combination with some of the list processing features of Logo. The microworld seeks to give the child the tools that will allow for future exploration regarding the meaning of language and ways in which words can be combined and used.

Exploring the World in Three Dimensions

This microworld focuses on creating and plotting objects using the Cartesian X, Y, and Z axes. It includes displaying three dimensional objects on a two dimensional screen, and rotating the objects in order to see the effects of rotation about a single axis. These rotations help upper elementary children concretize the three axes off the Cartesian coordinate system, particularly the Z axis. This is normally the most difficult axis for children to visualize. Rotational as well as other types of symetry are also considered in three dimensions.

Some of the other microworlds that were piloted as part of this program included exploration into poetry, discovering the structure of crystals, developing a data base on the chemical elements for a high school class, exploration using the Pythagorean theorem, and one that developed a "MacIntosh" style desktop for Logo in an effort to make Logo more user-friendly for some students.

Results of these explorations, the problems encountered and the learning that took place will be discussed.

Dr. Chris Templar
Johnson Bible College
Knoxville, TN 37998
(615) 573-4517

Towards Intelligent Microworlds

Wallace Feurzeig
BBN Laboratories
Cambridge, MA 02238

Logo microworlds do not teach. Like real worlds, they do not give away their secrets or explain themselves to passers by -- they simply exist and behave. They can be probed, their behavior can be explored, their structure understood, their underlying operation discovered. But, unlike CAI systems, "intelligent" or otherwise, they do not teach or even advise.

Microworlds are designed as learning environments. In interactions with a Logo microworld, a student strives to acquire or construct knowledge through active exploration and experiment. There are no programmed scenarios or built-in agendas. Concepts like lesson and instructional sequence are not a part of this experience, they are outside and separate. When microworlds are used in Logo teaching programs, microworld work is complemented by classroom activities.

Unfortunately, the central role of teaching within the total Logo educational paradigm is often ignored. This has given rise to a bizarre distortion of the Logo views of the essential contribution of teaching and the nature of microworld learning experiences. A large and influential segment of the education community attribute to Logophiles the following kind of wildly romantic and unrealistic educational perspective: that exposing students to a semantically rich microworld will spontaneously generate discovery and invention, promote the development of general thinking skills (whatever these are), and engender problem-solving skills that transfer directly to other task domains.

I don't think anyone in the Logo community actually holds such views (though some have come very close to implying them in the interest of arousing educational consciousness). The fact is that while CAI technology is often directed at teacherless teaching, Logo technology has always been committed to the social milieu of human interactions among children, teachers and computers. Within this context, Logo microworlds can contribute enormous learning benefits.

But it is also true that, without the aid of a teacher, many children do not learn in a Logo microworld. They are not able to set their own goals, to find effective methods of thinking about problems, or to acquire the skills of exploration, conjecture and inference. Left to themselves, they thrash about. They haven't learned how to function in an open learning environment. Skilled teachers overcome these deficits by providing the guidance and support that make microworld experiences productive for their students.

It is reasonable to ask whether the guidance and support needed by some students can be made an integral part of a microworld, and whether this can be done in a way that does not do violence to its fundamental character by turning it into a prescriptive system. Put another way, the question is whether microworlds can incorporate tutoring capabilities that are expressly designed to support exploration and inquiry learning and that are invoked optionally, if at all, at the request of the student.

This idea is beginning to be explored. It has given rise to the notion of an "intelligent" microworld, as dubbed by Patrick Thompson. It differs fundamentally from intelligent CAI systems (intelligent tutoring systems), in that its goals and operation are based on a learning rather than a teaching paradigm. The nature of the tutoring is different in the two kinds of systems. It is not only that the tutoring in intelligent microworlds is optional. Its goals are to aid exploration and inquiry rather than content acquisition, and its methods and operation are, like the microworld's facilities for exploration, non-prescriptive.

Intelligent microworlds need a domain expert, a program with capabilities for performing the operations in the microworld repertory and also for carrying out a rich variety of tasks like those addressed by the student. The expert has to be able to explain the purpose and use of microworld operations as well as its own task performance. Intelligent microworlds also need to have knowledge of standard student misconceptions about the meaning and effect of microworld objects and operations.

The expert serves as a kind of genie, available to the student on call. It has to be able to explain why the microworld responds as it does to any student command. It also has to explain its own operation when it demonstrates the performance of a task posed by the student. The student can also call the expert to test the student's work. The student may ask the expert to pose a problem, and to assess and evaluate the student's solution. The expert may be asked to critique the student's prediction of the outcome of a command whose effect is not what the student had expected. The expert will try to infer the student's specific difficulty from its knowledge of misconceptions.

Other kinds of capabilities are being explored within this emerging area of development. For example, Patrick Thompson has developed methods to support the acquisition of exploration and generalization skills within a mathematical microworld on transformational geometry (MOTIONS). I have developed an approach to aid the acquisition of planning and hypothesis testing skills in electric circuit troubleshooting within a microworld on elementary electricity (QUEST). The intelligent microworlds ideas and methods implemented in MOTIONS and QUEST, and those planned for FRAC, a new Logo microworld on fractions and fractals, will be described and illustrated.

Logo's Advanced List Processing:
A Vehicle for Natural Language Experimentation

Susan R. Harroff
Assistant Professor of Computer Technology
Indiana University/Purdue University at Fort Wayne

The philosophy of learning through experimentation which inspired the development of Logo and has since guided its use in the educational setting centers on direct use of the Logo language in solving a problem or carrying out a task and has typically involved the turtle graphics portion of the language. The list processing portion of the language may also be used as a tool for experimentation on two levels: Programs written in Logo may provide a context for experimentation involving areas other than movement and geometric shapes, for example the area of natural language. On a second level students may use Logo directly to develop procedures which demonstrate certain features of a natural language, thereby gaining a deeper understanding of the language by trial and error as well as by consideration of the problems involved in accurately representing a natural language in the more limited computer language. This paper briefly outlines a project based on the first level and suggests the addition of features which would make possible limited experimentation on the second level as well.

The project described here is an attempt to use Logo to create a microworld for second language acquisition analogous to the turtle graphics microworlds of mathematics and physics. The components of this language microworld are designed for experimenting with features of German and they are based on Logo's list processing rather than on its graphics capability. "Die Sprachmaschine"--the language machine--creates a microword in which students of German experiment with sentences which include one of a special group of prepositions--that is, with a grammatical concept which is typically difficult for students learning German as a second language to internalize. Students control this language machine using a limited portion of the German language. Machine responses reinforce the use of case endings and correct word order in German sentences and provide semantic checks for reasonable word combinations.

The environment created is experimental in that students may choose German words from "word bins" provided to them upon request and observe the syntactic and semantic processing that takes place on the chosen list of words. The sentence builder explains the process it follows in generating a completed sentence and informs the student if the word list contains incorrect word order or semantic errors. When processing is complete students may ask questions about the grammatical form of the sentence just completed, for example why a word in the sentence has the particular case ending that was generated. Students may also ask about structural and semantic information stored with vocabulary words "known" by the speech machine (gender or semantic category of nouns, type of verb, particular grammatical forms of a noun or verb). All such interaction between student and machine is in German so that the student engages in an experimental (written) conversation with the speech machine which may or may not produce the desired results, depending on the student's language choices.

A valuable level of experimentation would be added by encouraging the student, as his/her knowledge of German expands, to add other capabilities to the present program. Adding vocabulary would require including with each word the information necessary for syntactic processing and, more importantly, would involved the student in careful consideration of the semantic relationships among words already stored in the program. Likewise, the task of expanding the speech machine's present ability to handle only present tense verbs to include the capability of creating German sentences in past and future tense would stimulate thinking about the derivation of language forms as well.

Students may be encouraged to develop other language related programs -- sentence generators, syntax checkers, semantic checkers -- for a limited range of sentence types. In the process they are likely to refine their understanding of the structure of the particular language with which they are working as well as to expand their awareness of the complex interactions of language syntax and semantic structure in general.

**The project described in this article has been used for part of one semester with beginning students of German at the freshman level in the classes of Dr. Stephen C. Harroff, who served as language and learning design consultant for the project.

The State of Logo in Physics Education

Tom Lough and Linda Morecroft

This presentation reports on a survey sent to over 100 physics teachers at all levels throughout the world. The survey attempted to determine the relative frequency and depth of Logo use in physics teaching. The survey also queried the specific topics for which Logo was used, methods of teaching with Logo, and informal observations made on student attitudes, behavior and understanding. The results of this survey will be of interest to all science teachers, administrators, and others considering the use of Logo in their teaching.

Our report will also form the basis for further discussion at a meeting of the physics special interest group at Logo 86. We plan setting up a Logo materials development institute to further aid the integration of Logo into the physics curriculum.

Tom Lough
Piedmont Virginia Community
College
Route 6, Box 1A
Charlottesville
VA 22901

Linda E. Morecroft (lem@athena)
Project Athena
E40-342g
MIT
Cambridge
MA 02139

Proposal for Presentation
LOGO 86

Submitted by:

Judy Le Gallais, M.A.
Logo Computer Systems, Inc.

Nicole Michaud, M.A.
Director of Educational Research
Logo Computer Systems, Inc.

A Curriculum Oriented Approach to LogoWriter

Exposure to Logo in elementary and high schools has become widespread in recent years. However, its use is often removed from other school subjects. Logo graphics provide a concrete and meaningful context for experimenting with geometry concepts, animation, graphing and games. LogoWriter, which is an extension of Logo to include word processing, makes words and language more meaningful.

LogoWriter can be used as a tool in diverse subject areas. This presentation will focus on the use of LogoWriter to develop projects that relate to language arts, science, social/communication arts and math. Following are examples:

> A survey for a social studies project. The results can be represented in a bar graph.
>
> A time line of historical events with a description of each event that is recorded.
>
> A composition or branching story for a language arts assignment.
>
> A dice game that reinforces the concept of probability.

A demonstration of projects involving novice through intermediate programming skills will be given.

Beyond Turtle Graphics?

by Michael Tempel and Nicole Michaud

Turtle Graphics provides a "low threshold" entry point into Logo. "List processing" is often seen as an advanced topic that follows a turtle graphics introduction.

Usually, the activities that involve list processing (sometimes refered to as "words and lists") focus on natural language. Most people never thought about processing a list before working with Logo. List processing is the vehicle used to explore language.

Turtle geometry is easy to get involved with, in part, because it is intuitive and visual. By including word processing features in the Logo environment, language may be approached in much the same way.

```
REPEAT 5 [FD 100 RT 144]
```

displays a star, and shows it emerge as the turtle draws it. With commands such as CF, CB (cursor forward and back), SEARCH, COPY and REPLACE we can manipulate text with equally visible results.

```
PRINT [HOW ARE YOU TODAY]
```

displays a message on the screen.

```
TOP REPEAT TEXTLEN [INSERT "? CF]
```

moves the cursor to the beginning of the text and then inserts question marks between the characters of the message. (TEXTLEN reports how many characters long the message is.)

Removing a word from a list might be done this way:

```
CT
PRINT [ORANGE APPLE PEAR]
TOP
SEARCH "APPLE
CUT
```

The first line clears the text and displays the message. The cursor is moved to the beginning of the line. The word APPLE is searched for, highlighted and then removed with CUT.

Turning this into a REMOVE procedure we have:

```
TO REMOVE :WORD :LIST
CT PRINT :LIST
TOP
SEARCH :WORD
CUT
END
```

In contrast, using list processing to work with natural language presents obstacles for beginners.

An aspect of the low threshold of turtle graphics and the word processing approach to language is that beginning activities rely mostly on commands. Even elementary list processing often requires putting together several operations. A string of operations is hard to keep track of. To grasp the process underlying

```
PRINT SENTENCE PICK :NOUNS PICK :VERBS
```

you must think about the whole line. PRINT doesn't know what to print until SENTENCE tells it. SENTENCE first needs to hear from PICK and PICK, each of which have to look at their inputs.

Writing useful list processing operations often involves a level of programming well beyond the threshold of Logo novices. A REMOVE procedure might look like this:

```
TO REMOVE :WORD :LIST
IF NOT MEMBER? :WORD :LIST [OUTPUT :LIST]
IF EQUAL? :WORD FIRST :LIST [OUTPUT BUTFIRST :LIST]
OUTPUT FPUT FIRST :LIST REMOVE :WORD BUTFIRST :LIST
END
```

And while

```
PRINT REMOVE "APPLE [ORANGE APPLE PEAR]
```

will display the end result

```
ORANGE PEAR
```

on the screen, the process is not visible as it is with the REMOVE procedure that uses wordprocessing commands.

This is not to dismiss the value of traditional Logo list processing when working with language. This second version of REMOVE is, in fact, more generally useful than the first one since it can be better used as a building block of a more complex program. Our discussion is about thresholds, not ceilings.

Using LogoWriter, a version of Logo that includes word processing capabilities, we will present introductory language activities that use this approach. Preliminary results of work with elementary and junior high school children will be discussed.

Exploring language in Logo may not really be "beyond" turtle graphics at all.

For more information contact:

Michael Tempel
Logo Computer Systems Inc.
555 West 57th Street
New York, New York 10019

Nicole Michaud
Logo Computer Systems Inc.
9960 Cote de Liesse
Lachine, Quebec H8T 1A1

Collaborating with Teachers to Evaluate Logo Learning

by Molly Watt and Daniel Watt
Educational Alternatives
Gregg Lake Road
Antrim New Hampshire 03440

"The most persuasive reason for evaluation in education is improvement -- of the program, of the classroom curriculum and practice, of the child's school experience. ... Essentially we are aiming for better education for children. ... [This] forces us to clarify our ideas about what constitutes a good education in the light of our beliefs and experiences, and then to compare the resulting theory with practice. The nature and extent of the disparity between theory and practice will indicate directions to be taken in the effort toward improvement." (Brenda Engel, Informal Evaluation, The North Dakota Study Group on Evaluation, University of North Dakota, Grand Forks ND, 1977.)

1. The Need for Logo Evaluation

Computers are being rapidly integrated into schools in this country. With their numbers growing at a pace that seemed impossible five years ago, even to the most enthusiastic advocates of educational computing, the presence of tens of thousands of micro-computers in the schools, and the existence of a staggering variety of educational software packages have opened the way to opportunities for significant enhancements to precollege teaching and learning. But hardware and software can not, in and of themselves, bring about the educational benefits that we would all like to see.

The use of Logo provides a striking case in point. Logo was developed in the late 1960s and early 1970s under the leadership of Professor Seymour Papert at MIT, with substantial support from the National Science Foundation. Papert and his colleagues set out to create a learning environment that would simultaneously foster the learning of important concepts of computer science, and powerful, generalizable problem solving strategies, while providing opportunities for students as young as elementary school children, "... to do mathematics, rather than merely to learn about it." (Seymour Papert, Teaching Children to Be Mathematicians, Logo Memo # 4, 1971)

Since the early 1970s, the success of Papert's best selling book, Mindstorms, and the effective production and marketing of versions of Logo for microcomputers, paved the way for the use of Logo in tens of thousands of elementary classrooms throughout the country. We have been told by an official of Logo Computer Systems, Inc., that approximately 150,000 copies of Logo have been sold for the Apple II family of computers in the US and Canada.

--- Evaluating Logo Learning ---
--- By Molly Watt and Daniel Watt ---

With such widespread use of Logo, it is appropriate for educators, educational decision makers, researchers and the general public to be concerned about whether Logo is being used effectively to achieve the educational goals for which it was developed: especially increased student abilities in problem solving, computer programming and mathematical thinking.

In our work we have the opportunity to travel widely, visiting many classrooms and speaking with educators throughout the US and Canada. In classroom visits we often observe that students have learned only the simplest aspects of turtle geometry, without exposure to some of the deeper and more powerful aspects of the language. Teachers often request support in evaluating their students' work, and in finding ways to help their students get more out of their Logo experiences. Yet we often find that the teachers themselves are not aware of Logo's potential for learning important programming ideas or problem solving strategies, or how to tap it.

Our own observations, along with the concerns expressed by classroom teachers, and the interest demonstrated in response to recent presentations on Logo evaluation, lead us to conclude that classroom teachers need practical and effective methods by which they can evaluate and enhance critical aspects of Logo learning among the students in their own classrooms. Similarly, educational decision makers need access to evidence that Logo works in classrooms, in order to make informed educational policy decisions about the use of Logo.

2. Logo research to date

Currently available studies of Logo cannot yet provide the answers that educators are seeking. During the past few years there have been a number of experimental research studies about the effects of Logo experiences on student learnings. Most of these studies, however, leap over the Logo experience in an attempt to focus on external measures of generalizable skills and behaviors that students might learn through use of Logo.

One common limitation of many of these experimental studies of Logo is that they concentrate on external effects, ignoring students' actual Logo problem solving experiences. Most reports of such studies also tell us little or nothing about the Logo knowledge and experience of the teachers involved. There seems to be an unspoken assumption in many reports that Logo is some kind of packaged treatment for which effects can be measured, demonstrating no awareness that Logo allows for many different kinds of learning, and as with other educational innovations, outcomes vary significantly with the teachers' knowledge, experience and teaching objectives.

A much smaller number of studies focus on close observation and analysis of student behaviors in Logo classrooms or in more restricted lab settings. These observational studies tend to involve small numbers of students, and their reports often tell us a good deal more than the experimental studies do about the specific teaching objectives and methods used in particular settings. They tend to focus on specific students' learning experiences in some detail. While they show that Logo-using students do engage in mathematical problem-solving behaviors, they also reveal specific difficulties students experience in learning certain aspects of Logo.

An important direction in Logo research deals with what we call "critical aspects of Logo learning." These are aspects of Logo that have been identified as being important to successful Logo learning, by teachers and researchers who have been paying close attention to the difficulties encountered by novice programmers.

3. Identifying critical aspects of Logo

As a first step in evaluating Logo learning, we have to decide exactly what aspects of Logo to look for, in collaboration with classroom teachers, who have been using Logo for some time. We define critical aspects of Logo learning to be those which:

a) have been identified by teachers and researchers to be critical to students' effective use of Logo
b) are not usually acquired by students without some form of teacher intervention
c) are present (or available) in most logo learning experiences
d) are readily accessible to observation by classroom teachers
e) are subject to being influenced by timely teaching interventions

We will choose specific topics from the three interrelated areas that are at the heart of most students' early Logo experiences: introductory concepts of computer science, problem solving strategies, and the development of mathematical thinking. This leaves us with many specifics to investigate, and we'll mention a few of them here.

For example, in the domain of introductory computer science concepts we might look at such things as the development of effective strategies for naming procedures and subprocedures, and the use and development of procedures as "tools" for multiple purposes and as modules for use in different contexts.

In the domain of problem solving we might consider how students learn to make realistic choices of problems to solve, and how they refine problems to make them more tractable. The development of debugging strategies and the persistence of particular types of conceptual bugs such as "projecting intentions onto the computer," are certain to be important as well.

And in the domain of mathematical thinking we might look for such things as student use of variables, student involvement in formulating, testing and revising hypotheses in the domain of turtle geometry, and the development of generalizable theories and heuristics, such as the idea that repeating a fixed series of turtle commands always leads to a closed figure. We will also look for the conscious use of the turtle as "an object to think with", that is, putting one's self in the turtle's place in order to solve specific geometric problems.

4. The importance of the teacher's role

Many educators, inspired by reading Seymour Papert's Mindstorms, seem to have interpreted Papert's emphasis on "natural learning," "exploration," and "learning by doing," to mean that Logo can be learned by children simply by interacting with the computer, with a minimal amount of instruction. Papert's goal of putting the child in control of the computer has been taken by some to mean that the teacher should stand back and let the student discover the powerful ideas embedded in Logo entirely on her own.

In our own work, however, we have found the opposite to be true. We observe that Papert's vision of Logo as an open-ended learning environment in which learners (adults as well as children) can take a large share of the responsibility for their own learning, requires a teacher with a deep understanding of critical aspects of Logo learning, and a large collection of ideas for supporting student projects, and probing student understandings.

Therefore, our next agenda is to develop methods by which classroom teachers can sharpen their understandings of critical aspects of Logo learning, so that they can enhance the learning that occurs among their students. Ideally, we will be able to work collaboratively with a group of experienced Logo teachers and gather reliable information about the Logo learnings that occur in their classrooms. By observing what it is possible to achieve in real classrooms with experienced Logo teachers, who are knowledgeable about critical aspects of Logo learning and appropriate teaching interventions, we hope to create a basis for evaluation which will support the learning of all Logo students.

Evaluation of Computers in School: Some Issues

Gita Wilder and Edward Chittenden

Educational Testing Service

Princeton, New Jersey

This paper will examine evaluation issues that are prompted by the introduction of Logo (and computers) into the curriculum of the public schools; the paper will also describe specific approaches to dealing with assessment questions. The analysis of these issues will be based upon our observations in several sites and upon our experiences in documenting the development of several different programs where Logo is featured. The analysis will also draw from previous research in open education, in which parallel questions concerning evaluation were confronted.

Our discussion will take account of three levels of educational evaluation, each addressing somewhat different purposes but each, presumably, determining the ways in which computer experience becomes part of a total program. These levels are (a) classroom-based assessment, conducted by teachers for the purposes of evaluating student progress and providing assistance as needed; (b) school-based assessment, undertaken by faculty and others, in which evaluation centers upon curriculum and program characteristics rather than upon individual learning; and (c) district-wide assessment, initiated by the central administration for purposes of "accounting" to the Board and public, often revolving around surveys and/or standardized tests. At each level, the selection of evaluation methods tends to reflect educators' priorities and their conceptions of learning; thus, classroom-based assessment often proceeds from assumptions that are quite different from those implicit in district-wide testing.

We will illustrate some of the dimensions of evaluation issues with examples from our own work:

-Working with teachers and consultants, we developed a "Logo Learning Checklist" which asked teachers to estimate students' mastery of selected aspects of computer learning. The development of this instrument will be described and the reasons why it proved to be both a success and a failure will be discussed. While the process of rating proved valuable to the extent that it helped teachers focus upon selected elements of students' work, the instruments as such was cumbersome; perhaps more important, it raised questions about the validity of assigning global ratings when student learning may be quite context-specific.

-We examined standardized test results for students in the Computer School (New York) in an effort to see whether effects of computer activities might show up in instruments that are not designed to measure such effects but that are often used to monitor the impact of a program. Would time spent with computers have deleterious effects on other achievements? Was there any evidence to suggest carryover from computers to the traditionally measured areas of academic achievement? Results of this analysis will be described.

By way of conclusion, we will call for greater involvement on the part of the Logo community in finding ways to use computers for assessment purposes. Considerable work has already been accomplished in this community on computer documentation of individual learning styles and outcomes. However, in order to meet the needs of program- and district-assessment, we need strategies for obtaining and analyzing work samples across groups of students. Such data could supplement results from traditional testing schemes. Computers make it possible to "open up" response options beyond the ubiquitous multiple choice item. [We will describe one venture in which children gave their answers on a science test, in the form of lists.] Unfortunately, much of the current work bringing technology and educational testing together remains tied to models of education in which learning is presumed to be linear, and teaching, prescriptive. The challenge is to create assessment techniques that are commensurate with assumptions of teaching and learning that underlie many of the Logo projects.

The Formal Side of Logo

Dr. Michael E. Burke
Dept. of Mathematics and Computer Science
San Jose State University

The premise of this paper is that programming and problem solving are difficult enterprises, and that informal coverage and cutesy presentation of Logo in a course in which Logo programming is a primary ingredient is performing a great disservice to students. It promotes sloppy thinking and it promotes the Logo programming language as a toy. There is no substitute for hard thinking in problem solving and programming. There is no greater satisfaction in results that are obtained from hard thinking.

This paper describes the use of Logo in teaching concepts of computer science, mathematics and problem solving. While originally taught as an introduction to computer science at San Jose State Univ beginning in 1981, this material has served a wide range of purposes ranging from 8th grade computer literacy and high school computer science to preservice and inservice training of teachers as math specialists. The material covered in these courses is centered in understanding how Logo works. It is described in a serious and formal way so as to obtain a firm understanding of programming languages in general, understanding the processes created by programs as well as to explore the relationship between computer science and mathematics. The continuous thread that is interwoven in the details comes from Marvin Minsky's 1969 Turing Award Lecture in which he states:

> "To help people learn is to help them build, in their heads, various kinds of computational models."

The fact is that Logo readily admits various kinds of computational models while languages like Basic and Pascal admit only the most primitive ones. For example, suppose that the assignment statement is suddenly removed from these languages. The Basic and Pascal programmers will be crippled. The knowledgable Logo programmer equipped with the appropriate model will be able to go about his programming as if nothing happened. Why?..., because all the Logo programmer needs is recursion and the If statement for control.

In order to become an advanced programmer in Logo, one must have a reasonable collection of models of the language itself. The best kind of model is a mathematical one. Developing such a model is serious matter that requires critical thinking. There is no room for fuzzy reasoning. Unfortunately, there is no mathematical model that suffices for all of Logo but they do exist for a significant portion of the language. Such a model can be very useful in understanding the processes generated by Logo programs.

Programming is a difficult enterprise, and most of the difficulty in finding the solution to a programing problem comes not so much from the available features of a programming language but from our ability or inability to conceptualize the solution. We must accept the fact that no programming language is going to teach or be a substitute for critical thinking.

This does not mean that treating the Logo language in a formal fashion must be a dry, tiresome endeavor. Classroom informality and formal presentation of material are common occurrences and have been very successful in education for a long time. Logo is not an effective tool to use in a revolution in education. It is a tool that allows us to easily skirt much of the dirt that accompanies languages like Pascal and BASIC, and it requires knowledgable teachers to serve as guides.

An annotated outline of the course is presented below and will be elaborated upon during the talk.

I. Expressions vs Commands (statements)

- What is the difference between the following:

```
2 + 3
show 2 + 3
```

- How are expressions evaluated?

```
sqrt 9 + 16
2 = 3 = "false
(last [sqrt 100]) + 1
```

- Develop evaluation rules that describe the evaluation of an expression. This gives us a mathematical model (called the simplification model) that tells us how the value of a Logo expression is obtained.

- Other notations (prefix, postfix) for writing expressions.

II. Functions vs. Procedures

- Displaying a value vs. outputting a value

Example:

```
to sum.sq :x :y
   output square + :x square :y
end

to display.sum.sq :x :y
   show square :x +  square :y
end
```

-How does Logo handle the evaluation of calls to user defined functions and procedures? Extend the simplification model to cover this situation.

```
Example:  sum.sq 2 + 1 4
          => sum.sq 3 4
          => (square 3) + square 4
          => (3 * 3) + square 4
          => 9 + square 4
          => 9 + 4 * 4
          => 9 + 16
          => 25
```

III. Recursive Processes

```
to draw :d :a
   fd :d  rt :a
   draw :d :a
end
```

The existing model describes how non-terminating recursion is carried out. The student now has a model of recursion that is not associated with repeat or a computer.

III. Conditional Control

- Introduce If and terminating recursive processes.
- Extend the model to describe evaluation of If forms.
- We now have all the computational power that any programming language can have.
- Linear recursive processes, iterative processes, tree recursive processes.

IV. The Level Diagram Model describing how Logo works.

- This model closely resembles the implementation of Logo.

V. Global and Local variables, Scoping Rules

- Introduce the Logo assignment statement (make), local and global variables, dynamic scoping rules, mention other scoping rules that could have been used in determining the value of a variable.

- Illustrate problems that can arise when make is used.

- Illustrate that the simplification model cannot be extended to include the assignment statement.

- Summary: make adds pragmatic value, does not add any power to the language, and it destroys our mathematical model of Logo.

Insights

- A programming language can be a mathematical system.
- Proving correctness of programs is possible.
- Relationship of recursion to mathematical induction.
- Concept of function

Projects, Styles, and Techniques

Brian Harvey
Learning and Epistemology Group
Massachusetts Institute of Technology

Projects, Styles, and Techniques is the recently-released second volume in a series of books on computer science based on Logo. The series, *Computer Science Logo Style*, is intended to bring to the secondary school and hobbyist audience a particular point of view about computer science: the artificial intelligence view. This way of looking at computers is quite different from the more usual software engineering approach. The latter philosophy grew out of the particular part of computer science called analysis of algorithms. In this area, you are always dealing with a very well-defined problem, and are looking for the best way to solve it. Software engineers like to start with a formal problem statement, and then design a computer program to fit. They believe that the design process should be *top-down*; you should start with the overall structure and work down to the details. Their preferred programming language is Pascal.

In artificial intelligence, the problems are not usually so well defined. Starting with a vague problem statement like "develop a good strategy for playing chess," AI programmers can't begin with a rigid program specification. Instead, they build *tools*: program fragments that can be pieced together to form larger programs. The programming process involves writing code, testing, coming up with new ideas, and modifying the program interactively. This process is encouraged by an interactive language like LISP or Logo.

The structure of this series reflects my idea of the stages in which a computer programming enthusiast learns about programming. In the preface to the first volume I discuss the reasons why I choose Logo as the best language to learn, but these stages would exist for any language. The first stage is dominated by the need to learn the rules of a programming language. The first volume of the series, *Intermediate Programming*, teaches the Logo programming language while introducing some of the ideas in computer science that led to Logo's design.

The second stage is the exploratory one, dominated by the learner's desire to produce significant programming projects that are both useful and interesting. This stage can fruitfully last several years. The new second volume is addressed to a learner in that stage.

Finally, there comes a time when the learner gets bored with just writing more and more programs, and seeks a deeper understanding of the issues behind this practical work. The forthcoming third volume of this series, *Advanced Topics*, will address the needs of these learners by introducing them to some of the elements of university-level computer science, still in the context of Logo programming.

A Dual-Purpose Book

Projects, Styles, and Techniques is really two books in one. First of all, it's a collection of programming projects. As such, it is intended for a reader who has learned the technical rules of Logo programming and wants to put that knowledge to practical use. The emphasis is on getting the computer to do something fun and interesting. Each of the ten projects in the book is there because I thought I'd enjoy writing it myself, not because it fit some subtle pedagogic purpose. The projects are offered as case studies, as examples to inspire the reader's creative efforts.

At the same time, I *am* a teacher, and in the book I try to teach some ideas about programming technique and programming style. Often there is an easy way and a hard way to achieve a certain result, and you're better off if you know the easy way. Ideally, as a teacher, I would look over a student's shoulder and talk about the techniques that apply to his or her own projects. I can't do that in a book, and so instead I present some projects of my own and discuss them in the same manner. In this talk, I'll discuss some of the pedagogic issues that arise in the book, as well as demonstrating some of the projects.

About the Projects

There are ten projects in this book, grouped into five categories. The projects reflect aspects of my own character: I came to computers by way of an early interest in mathematics; my computing background is in artificial intelligence and in systems programming; I tend to think in words, not in pictures.

The first category, cryptography, includes a *Cryptographer's Helper* that assists the user in breaking a simple substitution cipher and a *Playfair Cipher* encoding project. The second category, games, includes a program that plays *Tic Tac Toe* with a perfect strategy and a *Solitaire* program that deals simulated cards and lets the user play by giving commands to move cards. The third category, mathematics, includes a *Fourier Series Plotter* that displays approximations to a square wave and a *Pitcher Problem Solver* that uses tree searching techniques to solve the IQ-test problems about pouring a given amount of water with different size pitchers. The fourth category, utility programs, includes a *Pretty Print* procedure that formats an arbitrary list for intelligible printing and an *Iteration Compiler* that translates iterative control structures like mapping into efficiant, tail-recursive procedures. The fifth category, pattern matching, includes a general purpose *Pattern Matcher* and an implementation of Weizenbaum's *Doctor* program that uses the pattern matcher as a tool.

Programming Techniques

The programming techniques discussed in the book range from simple ones like the use of *flag variables* (that is, variables whose value is always TRUE or FALSE) to more advanced ones like depth-first versus breadth-first *tree searching.* Down-to-earth questions like "When you have more than one end test in a recursive procedure, how do you know which to put first?" alternate with stylistic discussions about, for example, the use and abuse of dynamic scope.

In the talk I'll discuss briefly some of the topics that come up in several projects: stacks, combining output and effect, indirect variable assignment, and data representation.

Fostering Pre-Planning and Debugging Skills in Young Children Through Increasingly Complex Logo Microworlds

by

Rina Cohen and Esther Geva
The Ontario Institute for Studies in Education
252 Bloor Street West
Toronto, CANADA

Abstract: The current two-year project at the OISE combines on-going development and field testing of a graduated series of Logo microworlds for introducing young children to Logo, with an evaluation of the effects of regular in-class use of this series in grade two and four classes. An indepth study of children's learning processes and problem solving strategies within these environments is currently underway.

Rationale: In recent years, a growing number of educators and researchers have come to appreciate the crucial role of teacher support and instructional intervention when introducing children to Logo. This is especially true for young children who, in addition to teacher support, may greatly benefit from special software utilities and simplified versions of Logo to help them focus on basic concepts without getting overwhelmed by the complexity of the standard Logo language. For instance, one of the teachers participating in the project, described her doubts when first asked to participate in it, saying:

> Having just completed an unsuccessful attempt at creating a Logo environment in my grade-two classroom, I hesitated to become involved. My students and I had been equally frustrated in our attempts to use Logo, they because of their inability to master the skills necessary to produce the intricate patterns or pictures they wanted, I because of the difficulty of introducing and developing the concepts my students needed.

The current project is designed to alleviate some of these problems.

Design of the Microworlds: Based on previous research on Logo with young children, in the current project we have designed and programmed a series of four, increasingly complex Logo microworlds, intended to gradually introduce the basic Turtle Geometry and programming concepts to young children. Each microworld includes a modified set of commands and utilities focussing on a limited number of concepts.

Each microworld is accompanied by detailed teacher guides and learning materials. Special software utilities, such as a "dribble file" facility, (allowing the recording of all the child's interactions with the computer), as well as additional software tools and games for use by the children, have also been developed.

The development of these microworlds was guided by analysing prerequisite skills and cognitive demands of typical Logo tasks, and by breaking up the learning process into smaller, manageable units. Through exploratory, playful interaction, as well as goal directed activities, the child is allowed to gain sufficient mastery in one microworld before moving on to the next. The microworlds provide simplifed, and at at the same time, enriched Logo environments, allowing the young child to gain control over each environment without being hampered by many of the complexities inherent in the full blown version of Logo.

The following principles and techniques have been used in the design of this series of microworlds:

(a) Each microworld forms an extension of the one preceding it in the series.

(b) The screen is bounded (that is, there is no wrap-around) and is viewed as a "magic room" within which the turtle can move and draw. To encourage use of measurement, the screen boundaries (or "walls") are visually represented as rulers, (without numbers), marked with visible units of distance, or "turtle steps" (each equal to 10 regular Logo units). When the turtle bumps into a boundary, an appropriate error message appears (e.g. "turtle hit the ceiling").

(c) Since young beginners often experience difficulty with the RT and LT commands, we start them off by exploring turtle movement in an absolute frame of reference (i.e. the turtle never changes its heading and can only move up, down or sideways). Only with the fourth microworld (called "Peter Pan Turtle"), is the child introduced to movement in a relative frame of reference (i.e. movement with respect to the turtle's current heading) as used in Turtle Geometry.

(d) The first microworld (called "Magic Room"), is "non-numeric", namely, no numeric inputs are used with commands. All movement commands are associated with a single "turtle step" (e.g. "U" for moving up a single unit). When the need for numeric inputs arises (as noticed when the child starts chunking sequences of identical movement commands on one line in order to move the turtle to a specific position on the screen), the second "numeric" microworld (titled "Turtle the Count") is introduced (here the "smart" turtle can count the number of steps it is told to move).

(e) To allow the young beginner to create meaningful and interesting designs within a relatively short period of time, three basic ready-made shapes (square, circle and triangle), colour filled or hollow, are available. The size of these shapes is fixed in the first non-numeric microworld, but has to be defined via numeric input in subsequent microworlds.

(f) To allow young children to comprehend and make meaningful use of the REPEAT command, an interactive version of this command is introduced in the third microworld ("Turtle Factory"). This simplified Logo environment enables the child to generate <u>pre-planned</u> recurring patterns, without having to deal with the added complexity of RT and LT turns within the repeat command.

(g) To facilitate comprehension of angle rotation in the "Peter Pan" microworld, students are introduced to non-numeric (i.e. fixed) right and left turns of 90 and 45 degrees prior to exploring numeric angle inputs. In addition, the turtle's pivotal movement occurs in slow motion. A set of games is also available to reinforce the concept of angle rotation.

<u>Pre-planning and Debugging</u>: In this section we would like to briefly outline how

pre-planning and debugging skills that are fostered in the current project. Children participating in the project are encouraged to develop and improve their pre-planning and debugging skills in direct mode as well as through a variety of programming activities. An example of early signs of pre-planning and debugging in direct mode was observed in a grade two class where a child explained that he was going "to make a 'spaceship' going into a black hole". (One should note that this objective was initiated by the child a short while after the explosion of the "challenger" in the United States.) The child who had been exploring at that time the "Turtle The Count" microworld started by changing the background colour to black, proceeded to create (in white) a series of increasingly larger circles with a common starting point, and finished by creating in solid orange a "spaceship" constructed from the ready made shapes, and placed towards the center of the "hole".

The simplified enviornments allow children to engage in pre-planning and debugging in slightly more structured direct-mode activities as well. For instance, appended to each microworld is a series of "challenges" differing in type and in level of complexity. Some challenges require the child to reproduce on the right a picture presented on the left of the screen. Others require the children to look at two adjoining drawings on the screen: a given drawing and a faulty reproduction of the drawing. The child's task is to indentify the mistake(s) in the reproduction, to write down on paper the commands which will "fix up" the mistake(s) and to try out their debugging on the computer, or, as one grade four child said:

"We have to debug our debugging."

Other tasks may involve making a design or a picture on a page which looks like the walled screen, write the commands that will produce the picture and then try it at the computer.

Such activities, implemented within the simplified microworld environments, allow the children to focus on various aspects of incresingly complex patterns without having to deal simultaneously with too many concepts. We have observed a child, who, when trying out her program at the computer noticed that she had made a mistake. She immediately proceeded, without any support from an adult, to correct that particular mistake on the program sheet as well as all identical "bugs" which appeared later in her program. The point we want to emphasize is that because she engaged in meaninglful programming that was appropriate for her developmental level, she knew, in advance, where her mistake was likely to reappear. These, and other similar observations are examples of how within the simplified environments described above, young children can develop their pre-planning and debugging skills and engage at the same time in meaningful activities.

A Developmental Sequence of Logo Activities

Logo is one of the most important components of the computer education program which is being developed by the Jefferson County Public Schools. The goal of this presentation is to describe and demonstrate some of the activities we have used to introduce Logo to students at all grade levels.

Classroom teachers and members of the school district's Computer Education Support Unit have developed a sequence of Logo procedures which allows students to begin using single-keystroke, pre-Logo activities and gradually progress into more and more complex activities. The following chart lists the names of the activities and the grade levels in which they are used.

Name of Activity	Grade Level
Kindergarten Logo	K
Shapeworld	K-1
Super Shapeworld	K-1
Turtle School	1-2
Slow Turtle	3-5

Descriptions of each of these activities follow.

- Kindergarten Logo

Kindergarten Logo is a single-keystroke activity designed for use by beginning Kindergarten students or by handicapped children who have difficulty using the keyboard. Keys on the right side of the keyboard turn the turtle right 15 degrees; keys on the left side turn the turtle left 15 degrees. The space bar moves the turtle forward 10 steps; the DELETE key clears the screen. Students who use this activity are allowed to create simple designs and might be asked to draw simple geometric shapes such as squares and rectangles.

- Shapeworld

Shapeworld is another single-keystroke activity and is similar to Instant Logo and EZ Logo. Shapeworld contains 5 functioning keys - F, R, L, J, D. F moves the turtle forward 10 steps; R and L turn the turtle 15 degrees right or left. J makes the turtle jump (Penup, Forward 10, Pendown). D is short for the Terrapin Logo command DRAW. This activity is used by Kindergarten students who have some letter recognition skills and by other students who are just beginning to learn about Logo. Students who are using Shapeworld are able to find the dimensions of the screen by counting the number of times F must be pressed to move the turtle from one edge to the opposite one. They are also able to use the J key to draw figures which are not connected.

John Watts
Computer Education Support Unit
Jefferson County Public Schools
4409 Preston Highway
Louisville, KY 40213

- Super Shapeworld

Super Shapeworld is an enhanced version of Shapeworld. The keys F, R, L, J, D function as they do in Shapeworld. There are 9 additional keys which have an effect on the turtle. The following list summarizes the function of these keys.

B	Box (draws a 20 x 60 rectangle)
S	Square (draws a 20 x 20 square)
T	Triangle (draws an equilateral triangle whose sides measure 20 steps)
C	Circle (draws a circle whose diameter is 20 steps)
E	Erase (erases the line drawn by pressing F)
2	Draws a 2x2 rectangular dot array
3	Draws a 3x3 rectangular dot array
4	Draws a 4x4 rectangular dot array
5	Draws a 5x5 rectangular dot array

The primary purpose of Super Shapeworld is to give students more tools to use in designing computer drawings. Super Shapeworld also allows students to begin to explore some of the properties of simple geometric shapes. Students can use the dot arrays to help them plan their designs. This activity can also be used to enhance problem-solving and pattern recognition skills by having students reproduce designs made by the teacher or by other students.

- Turtle School

Turtle School is an activity designed to help students make the transition from the single-keystroke activities to Logo. It is used primarily by 1st and 2nd grade students who might have difficulty understanding numbers greater than 100. The most important commands are F, B, R, L which correspond to the Logo commands FD, BK, RT, LT. Each of the commands requires an input. The major difference between the Turtle School commands and Logo commands is the size of the numbers used. The numbers used in Turtle School are smaller than those used in Logo. For example, the Turtle School command F 5 is equivalent to the Logo command FD 50; R 6 is equivalent to RT 90. The transition from Super Shapeworld to Turtle School may be made by noting the number of keystrokes required to make various designs. For example, a student who draws a line segment in Super Shapeworld by pressing the F key 5 times may draw the same line segment in Turtle School by typing the command F 5.

Turtle School also contains procedures which correspond to the other keys used in Super Shapeworld. The following list shows the relationship between the two activities.

Super Shapeworld Key	Turtle School Command
B	RECTANGLE
S	SQUARE
T	TRIANGLE
C	CIRCLE
2	DOTS2
3	DOTS3
4	DOTS4
5	DOTS5

It is important to note that Turtle School adds commands to regular Logo. All Logo commands may be used along with the Turtle School commands. In particular, the REPEAT command may be used, and procedures may be defined.

- Slow Turtle

Slow Turtle is the final activity in the sequence. Students usually begin to use this activity in 3rd, 4th or 5th grade. The turtle movement commands are F, B, R, L, each of which requires a single input. The only difference between these commands and the Logo commands FD, BK, RT, LT is the speed at which the turtle moves. For example, the Slow Turtle command R 90 causes the turtle to turn right 90 degrees, but the turtle turns much more slowly than it does when RT 90 is used. The primary reason for using the slow commands is to help the student visualize the effects of the turtle's turns.

The transition from Turtle School to Slow Turtle may be made by discussing step sizes and turn sizes with the students. In Turtle School, the turtle takes "giant steps" and makes "giant turns". In Slow Turtle (and in regular Logo) the turtle takes "tiny steps" and makes "tiny turns". The following chart illustrates the conversion factors between the two activities.

Turtle School	Slow Turtle
1 giant step	10 tiny steps
1 giant turn	15 tiny turns

Procedures for drawing the simple geometric shapes and for making the dot arrays are also contained in Slow Turtle. The names of these procedures are the same as those in Turtle School.

The remainder of this presentation will be devoted to describing lessons which have been developed using the Logo activities described above. Participants will receive copies of the procedures used and of various worksheets and lessons which have been developed.

LOGO in Quebec: A Report on the "Revolution"

by
Pierre Bordeleau and Laura R. Winer

APO QUÉBEC
*Centre québécois de recherche sur les
applications pédagogiques de l'ordinateur*
Montréal, Québec

Quebec had the Quiet Revolution of the sixties and the not so quiet revolution of the seventies, but what of the Logo revolution? The impact of *Le jaillissement de l'esprit* was comparable to its English counterpart Mindstorms. Logo was not going to be just another piece of educational software, nor just another programming language. A French version of Logo was created, which not only facilitated its entry into the Quebec educational scene, but made it doubly welcome as one of the small number of pieces of educational software available in French.

Logo did not come into Quebec schools alone, however. It was part of an overall Microcomputers Implementation Plan, first introduced in 1983, a revised version of which was submitted in 1985 with a time-line until 1991. This plan outlined in some detail the direct impact of micros for the elementary and high school curricula, as well as for teacher and administrator training. It also outlined the needs in the areas of development and production of software and courseware, and research.

Logo was specifically mentioned in this plan as one of the potentially creative applications to which elementary school students should be introduced. At the same time, however, no change in the current curricula was perceived as necessary. Obviously, at least the administrators felt that Logo could be incorporated with minimal change to the system and still benefit the students.

But what do the educators think? How has Logo been received in the schools? And where is it going? These are some of the questions which we will examine in our review of Logo in Quebec. Logo has been the object of intensive study by researchers here: numerous projects with concerns ranging from the impact of Logo on problem-solving, mathematics, and language skills; the use of Logo for linguistic analysis; the effects of grouping when children are working with Logo; new micro-worlds--in the social sciences, for example-- have been created using Logo; and the age of those playing with the powerful ideas has been raised to encompass college and university level students.

Logo has also found its way into the classroom, and as a much more general tool than just for the teaching of mathematics. Logo is being used for word processing, language skills, general problem-solving, computer literacy, and programming.

Logo is also the introduction for many teachers to the whole area of educational computing, whether in their schools, in informal exchanges with other teachers, or in more formal workshops or courses offered by their school boards, the universities, or the government.

It is clear that to some extent a "radical change in the social structure of learning" has occurred and will continue to evolve; it is also clear that this will continue to be an evolution rather than the revolution envisaged in Mindstorms. But this does not mean that the influence of Logo as not only a programming language but also as a philosophy will not be felt to some extent in Quebec. Where that evolution has brought us and where it will lead will be the subject of this paper.

Several innovative applications of Logo in the classroom will be highlighted; as well, the "common or garden" variety of uses will be discussed--the real impact of any innovation must be assessed by examining not only the exceptional cases. As indications of what the future may hold, ongoing research projects involving Logo--at all levels of the educational system--will be reviewed.

TEACHERS TRAINING IN LOGO IN SPAIN: THE ROLE OF LOGO IN THE OFFICIAL PLANS OF THE SPANISH EDUCATION MINISTRY

(Paper abstract to be presented in the LOGO 86 Congress in Cambridge ,Massachusetts)

This paper describes the role of Logo in the Atenea Project (the ministerial project in Spain dealing with the introduction of computers in the state educational centers). It will deal with the teachers training plan, its methodology and with all the official activities to promote the development of Logo in this country.

Atenea Project

The aim of this project is the rational introduction of the new information technologies in the state educational centers as a way to improve the educational system and to put it in accordance with the today's society. The main aims are:

- -To develop in teachers and students the abilities to create, select and process information.
- -To use the new technologies in order to improve the teaching process in all the curriculum areas in which computers can contribute carrying out tasks that cannot be achieved through other media or improving the traditional ways of carrying them out.
- -To introduce new topics in accordance with the new needs which our information society has.
- -To use computers to create new learning environments in order to develop creativity, self-esteem, autonomous learning and thinking processes in the students.

Teacher-trainers plan

The necessary teacher training plan to reach the above mentioned goals is conducted by the Institute of Technology for Education (ITE), and is divided in two stages:
the first one deals with the training of 56 teacher-trainers who will conduct the training process in the Teachers Centers, one in each territorial area. This stage was carried out last year and had a duration of a whole academic year. The formation supplied covered different topics related with the computers curriculum in their instrumental aspects and their educational uses: programming languages and their didactic topics related to them (specially Logo), data bases, word processing, electronic spreadsheets, computer aided instruction, simulation, games and so on. I will describe later all the topics concerning to the role of Logo in the whole plan.
In the second stage, the above mentioned teacher-trainingers

organize several courses for other teachers, those who will use computers in their classrooms. These training courses begin with an introductory curriculum on computers and education covering the main topics:
general aspects, word processing, data bases and (mainly) Logo. After this short course, the teachers in training are separated in their respective interest areas and each topic is covered in depth separately: word processing and data bases for humanities teachers, Logo for those who will teach computer literacy or will perform curricular integration in any area (mainly Mathematics and Physics), processes control for technology teachers and so on.

Initial steps

The initial steps began with the implementation of several research works dealing with teaching Logo to adecuately selected children and teachers, at the same time the ITE built a more thorough course "Logo and Teaching", that is completed by 300 teachers. As a result of the above mentioned preliminary actions, a need was felt of defining a standard version in Spanish language covering all the primitives before beginning the project itself. In order to reach an agreement the ITE organized an open meeting with experts working in Logo in Spain and the main software developpers. As a conclusion of the work sessions the expected agreement was reached and at the present moment there are several implementations of this standard version for different computers.
The ITE ellaborated the training plan for the Atenea Project Teacher-trainers which was designed to last for a whole academic year.
The topics related to Logo (objetives, methodology and contents) will be described in more detail.

The role of Logo

The topics covered can be divided in the next main groups: didactic foundations, instrumental knowledge, methodology of teaching, analysis of experiences, integration in the curriculum and research techniques to be applied in the classrooms.

The contents try to cover in depth all the topics: educational philosophy, turtle graphics (including 3D designs), words and lists, modular and functional programming techniques, desing of applications, microworlds, problem solving foundations and Artifficial Intelligence. Here is a brief description of the instructional methology used: there are two teachers for each computer (MS DOS compatibles), small group sessions are held and some didactical applications (some of which are open) are proposed to the

participants in order to stimulate small group discussion about their educational value, once a given application is selected, it is carried out as a problem solving activity in the small groups. The groups must include remarks about the goals to be achieved in the classroom and the proposed methodology. All the groups present the job done to the whole group. This activity gives way to a new discussion about the educational possibilities of Logo. Thus, the educational Logo approach is assumed in a natural way.
All the applications developped by the groups become new material for the course documentation for later experimentation with pupils. There are also many models in the documentation covering different topics for later use in the classroom: learning environments in propositional logic, structured programming, problem solving in several areas and many others showing the Logo educational power.
At the same time sessions are held in order to analyze different experiences based on the Logo approach in all the educational levels which have been carried out by other teachers. Research techniques and observation systems are also presented by specialists in pedagogy and psychology.

Final remarks

The Logo phenomenon has increased quickly in the last two years in Spain, there are two magazines specially devoted to Logo, an association of Logo users, many books, articles and a great amount of experiences. The above described plan undoubtly will represent an official impulse that will achieve a qualitative leap in the next years. About 3000 teachers are going to be trained in Logo during the present academic year and the results will be seen in the next one.

Luis Rodríguez-Roselló

Head of the Institute for Technology in Education Research Departament. Alcalá de Henares, Madrid (Spain)

Louisa Birch
Meadowbrook School
Weston, Mass. 02193

THE PUPPET THEATER

Each morning in the Kindergarten classroom the children choose activities from a variety of materials. Some build in the block corner, some play pattern, matching, sorting and number games at the math table, others are up to their elbows in glue and paint at the art table. Three or four are busy at the sand table and a few more are happily creating and acting out a story in the house corner. Others are using one of the two computers. One machine has a Logo program which enable the children to easily guide the turtle to draw on the screen or else one which enables them to direct the floor turtle to visit various spots in their child created "farm" in one corner of the room. The other computer has a new and wonderfully interactive language arts program called PUPPET THEATER. This program is written in Sprite Logo and uses a voice synthesis called Votrax so that the children not only see what they type on the monitor, but they also hear it.

On this morning Ann and John choose to use the PUPPET THEATER. Ann decides to have one of the puppets, Ed say " It is spring I dont like winter". In response, John has his puppet Fred, say "I like to play football when its warm". After each child types in a sentence, the Votrax "says" whatever has been typed. As it says the words, they blink on the screen so that the child can easily follow the test as the words are spoken. When the Votrax has finished talking, the puppet's eyes roll appealingly. A cardboard keyboard insert is put on the machine above the numbers to let the children know which keys to use for various functions. They may repeat a sentence they have just typed, save it, switch to the other puppet and type something new. They soon learn to read the words on the insert and remember which key to press for the function they are ready to use. John likes to hear the puppet speak, so he presses the repeat button several times before he saves his text and gives Ann a turn.

Frequently the children have no idea how to spell the words they want to use. Different children approach this problem in different ways. Some, the ones who are cautious and fear making mistakes, ask for the spelling of almost every word. Instead of simply telling them the spelling, the teacher writes the word in a class dictionary. The dictionary is a little notebook with the edge of the pages cut away so that A, B, C, etc. may be seen down the side of the pages. After a while the children learn that rather than asking for the spelling of a word they can look under the letter it begins with in the dictionary and try to find it themselves. If it is not there, or if they cannot find it, then they ask to have it added to the dictionary.

Although at first the dictionary was used by those children who were the most cautious, soon all the children realized it was a useful resource and began to use it. In addition to using it to look up the spelling of words, some children use it to get ideas of what to write about. They browse through the pages until they find a word they recognize and that interests them and then they use it in a sentence. A sample page in the dictionary is as follows: come, cookie, costumes, changed, Chinese, chickadee, Christmas, chicken, cabbage.

Other children use a more experimental or inventive approach to their spelling. They simply spell the words the way the words sound to them. This is a conversation two children made later on in the morning with the puppets:

Ed: Do you wat to cum over to Mi hous

Fred: Yes I do

Ed: How do u like my hairdo

Sometimes the misspelled words are phonetically accurate and thus correctly pronounced by the Votrax. Others are mispronounced. When the Votrax pronounces a word differently from what the child expected the child is usually surprised and then bursts into laughter over the funny way the machine said the word. Occasionally the child will seek the correct spelling and then rewrite the sentence. Usually he simply continues writing.

Much of the learning comes when the children work together. They eagerly discuss what they will write about, how they will phrase their ideas, what words to use and how to spell the words. Naturally some children work better together than others and it is important for the teacher to encourage children whose learning styles are similar or whose learning styles will compliment each other to work together. Sometimes a child who is really beginning to read and write works well with another who barely can sound out the simplest words but who has a wonderful imagination. The reading and writing ability of one child in combination with the imaginative ideas of the other can often produce creative work. At the same time one child is learning to become more imaginative while the other is learning to put her thoughts and ideas into written words.

After each child has written a sentence he saves it and then at the end of the morning the children direct the machine to "say" all that has been written that morning. As with each individual sentence that is written, the computer shows the text and the words blink as they are "spoken" so that the children may easily follow the words. Next the entire text is printed out and each child who has used the machine during the morning is given a copy of the printout. The children eagerly read through it and identify their own words and then try to remember who wrote the other sentences. They read the words to each other and together with great pride. This is their own writing and that of their friends. They want to read and reread it again and again. It is especially nice, when writing by hand is still difficult, to have one's own letters formed neatly and printed out so nicely. Of course, it is much easier of read a friend's writing also.

The most common way to encourage children to write with the computer is to put a word processing program into the machine, assign them a topic or give them some questions to answer and ask them to write. This is probably an effective method for those children who really have something to say and who like to write. Unfortunately writing is really hard for many children. They have few role models to follow as they rarely see an adult write. Often their writing is corrected and criticized so that they are inhibited from writing freely. Many children who speak easily simply "don't know what to write" when they are asked to put their thoughts on paper. Even when their is plenty of positive reinforcement and good help, the words sometimes come painfully slowly and are often awkward. The PUPPET THEATER helps to make the writing process more natural. The children have the puppets talk with each other just as they might talk together. This is familiar and they are better able to come up with ideas because it is more like a conversation. It is easy for the children to help each other for as the puppets interact so must they. Although, at the kindergarten level, one does not find real story writing, it is evident that the children are learning to enjoy writing just as they enjoy speaking. They see the written word simply as another form of communication and they begin to write for the same reasons they speak: to share their ideas and thoughts with others.

The Votrax allows the child to hear what she has written. It gives reinforcement and non-judgmental corrections which the child may choose to accept or ignore. It also enables the child to use another sense when writing. The child who learns better auditorily rather than visually has his strength reinforced. Having the Votrax orally "read" what the child has written also encourages the child to reread the text himself. This is extremely important for a good writer to do, yet it is often very difficult to get children to reread their own writing. If they get in the habit at this young age, perhaps they will do it naturally when they are older. The more they read their writing the more they will rewrite it and the better writers they will become.

Most important of all, the children really enjoy using the PUPPET THEATER program. They eagerly await their turn at the computer, they have great fun thinking up things for the puppets to say and they are anxious to read the final printout. This is indeed a creative and imaginative way to introduce children to writing and reading.

THREE DIMENSIONS OF SPEECH WITH LOGO

Author : **D. J. Cartmell**

Date : February 24, 1986

INTRODUCTION

Have you ever used a talking program? Some people use them all the time, e.g. the blind. Have you ever talked to a program and had it respond either by action or with a voice? Not as many people have done this, but it is possible! We are finding that in the field of education, the most effective program systems are those incorporating multiple modality presentations. That is, the creative use of color, graphics and sound are the basis for highly effective teaching tools. Such systems appeal to the eyes, ears, and touch as well as stimulating the cognitive process. And how about LOGO?! Can voice and sound be used in LOGO programs? The answer is a loud "YES".

This paper shows how voice can be used with LOGO. In fact, it explores three different dimensions of voice and shows how these can be used with LOGO, i.e. digitized speech, synthesized speech and recognized speech. To demonstrate that these types of speech can all be used, the presentation includes live examples of all three utilizing the IBM Voice Communication Option with IBM LOGO.

BACKGROUND

The IBM Voice Communication Option is a card for the IBM PC family which includes the following functions: - Digitized Speech - Synthesized Speech - Recognized Speech (or Speech Recognition) - A Built in Modem supporting Telephony Functions. A support facility provided with the card is an Application Program Interface program. This allows programs to interface directly with functions of the card. For example, LOGO can use this interface to access the various speech functions.

DIGITIZED SPEECH

Definition

Digitized speech is similar to speech recorded with a tape recorder. Thus you use a microphone with voice as input and a speaker or head phones for voice output. However in the case of the computer, the analog sound waves are converted to a digital form through a sampling process. Instead of recording onto tape, the computer records digits representing the sound waves onto a disk. To play back the recording, it is necessary for the computer to convert the digitized form back into the analog wave form of the original speech. Since the sampling process takes place at a high rate, a good fidelity recording can be achieved.

A First Example

(Recitation of a poem...both prerecorded and recorded live at the time of presentation by a member of the audience.)

Features...Considerations...Limitations

LOGO Interface

The LOGO interface includes descriptions of the API Interface. It also includes a description of the programming interface where a LOGO procedure "SAY.BYNAME" is used to playback selected recorded phrases out of a directory. It includes record and playback interfaces and a file subsystem for voice.

Other LOGO Examples : Graphic programs and arithmetic program examples.

SYNTHESIZED SPEECH

Definition

Synthesized speech is often recognized as robotic speech. A synthesizer accepts any text data stream and attempts to voice it. It uses a series of rules and algorithms to determine the voicing. The more robust the rules and algorithms, the more natural the speech reproduction.

A First Example

(A simple arithmetic drill and practice program which uses synthesized speech for word problems.)

Features...Considerations...Limitations

A text data stream usually contains much more that character strings separated by blanks. It also contains numbers, punctuation, capital letters, abbreviations, acronyms, columns of words and numbers as well as other constructs. All of these must be handled by the synthesizer. It may be important that the application program also be able to control this level of processing. Of course the other obvious consideration is correct pronunciation. How can the program or the user control how the synthesizer actually pronounces the words.

LOGO Interface

The LOGO interface includes descriptions of the API Interface. It also includes a description of the programming interface where a procedure "SAY" is used to voice character strings. It describes how the interface allows for the control of the various features and switches of the synthesizer.

Other LOGO Examples - Graphic programs and arithmetic program examples.

RECOGNIZED SPEECH

Definition

Recognized speech or speech recognition is different than the other two types of speech described earlier. In the first case, you started with voice as the input and you got voice as the output. In the second case, you started with text as the input and got voice as the output. In the case of recognized speech, you have voice as the input, but the output is a process or a function which is activated. That is, you have a voice as input and an action as an output. Recognized speech involves a three step process. First you choose a set of words which you want to be recognized. Second you define a set of processes you want to action when the words are recognized. And thirdly, the user goes through a training phase to train the machine to recognize their particular voice.

A First Example

(Drawing a simple figure using LOGO commands recognized by the computer.)

Features...Considerations...Limitations

It is usually considered a limitation that the recognizer can have only a few words available for recognition at any one time due to the amount of space required to hold word voice models, e.g. some systems are restricted to 24 or 64 or 128 words. In the case of the VCO, new sets of words can be brought into active status dynamically by setting up a tree type structure for the words. Thus when one word is recognized, it can cause a new set of words to become active which replace the original set thus effectively allowing an open ended vocabulary.

LOGO Interface

In the case of recognized speech, there need not be any direct interface into the LOGO program since a background utility program comes with the VCO card which performs the recognition function. Where the interface comes is in defining LOGG commands or procedures which are associated with words which get reccgnized.

Other LOGO Examples -

Digitized speech programs and synthesized speech programs will be run having been activated via voice recognition. Thus it is possible to have voice as input and the action effect a procedure generating voice as output.

CONCLUSION

The various dimensions of voice technology provide new avenues for exploring with LOGO. For example, now when children create procedures with LOGO, they can add their own voice track co their procedures. The speech capability provides that 'other' modality of audio which is so necessary in educational systems.

EDUCATION

SYSTEMS

From : David J. Cartmell
Special Education Projects
Education Systems
D43H/949
Kingston, N.Y. 12401

Logo in the High School Computer Science Curriculum
What Constitutes Success?

Sharon Burrowes
Wooster City Schools
Wooster, Ohio 44691

In the few short years since microcomputers have entered our schools, the computer curriculum has become increasingly rigid: Logo in the elementary schools, Basic at the Junior High, and Pascal at the high school. Those of us teaching in the public schools find ourselves contending with formal curriculum guides, standardized "computer literacy" tests and Advanced Placement Exams. In this rigid environment, it is easy to take the view that the student who masters all the objectives or scores well on the standarized test is the one who is successful.

Ever since we introduced a Logo course into our high school curriculum, it has been very clear that there is a difference between students learning Logo as a first language and those learning Pascal. Students learning Logo are more exploratory and experimental. Further, a broader range of students get excited about Logo. However, there are still students who, using the traditional measures, fail the course. Do these students also "fail" Logo? Each semester that I teach this course, I find myself asking more questions about what it means to succeed in this course. Further, I have become increasingly curious about what factors contribute to "success".

In the fall of 1985, I decided to collect some data on each student along with samples of each student's programming with the goal of better understanding what my students were thinking, learning and feeling. The data collected fell into four categories. First, I used four previously validated items from a paper and pencil test of Piaget level, An Inventory of Piaget's Developmental Tasks.[1] The items used were volume, rotation, classes, and distance. This test was given just before Logo was started then again at the end of the semester course.

Second, I borrowed from an earlier study an attitude measuring instrument which assessed primarily enjoyment of computing, self concept in computing, and anxiety towards computing.[2] This instrument was given just after turtle graphics was introduced, again just after procedure inputs were introduced, and finally at the end of the course, immediately after several weeks of introductory list processing.

Third, grades on the thirteen small programs that were assigned during the course were averaged. These scores, which correlated highly with the final grades in the course, provided a measure of success as traditionally measured in high school courses.

Last, three samples of each student's programming were collected. These were in the form of short "quizzes" which were given to students in such a way as to encourage their best effort. The first and third quiz each presented students with different designs made up of squares. They were asked to write a program which would produce the given designs. The second quiz was more traditional, based on questions that had been asked in other testing situations. The questions on this quiz included the use of inputs, recursion, and the use of interactive programming. The first quiz was given shortly after inputs were introduced, but before students had been assigned a program using them; the second was given about half way through the Logo unit; and the last quiz was given at the end of the course.

When all the data was collected and compiled, the results, both observational and statistical, made me rethink just how I would define success in this course. Of course, a school system would say that that students who passed the course were "successful". However, examination of the programming samples, especially the first and third quiz indicated that nearly all students were able to write some sort of Logo program.

The responses on the quizzes varied widely in accuracy. What seemed more significant, however, was the fact that programs fell very cleanly into about five categories which might be characterized as very concrete to very general. There were both "correct" and "incorrect" programs in every subgroup. Perhaps "success" could be related to generality of Logo programming.

However, the data I collected showed that there were a number of students who could write Logo programs in a fairly general manner who failed the course simply because they turned in little, if any, of the assigned work. Were these students successful in the course...and, if so, should they have received failing grades?

Suppose, then, than generality were used as a measure of success. Perhaps generality in programming is related to Piagetian level. Maybe the students who could program in a general manner had achieved formal operations while those less successful had not. If that were the case, then using generality as a measure of success would grade students on something over which they have no control. Neither statistical tests nor examination of the work of individual students bore out this hypothesis. There were students who scored "formal" on the Piaget assessment whose programs were very concrete in style and students who scored "concrete" who were able to produce fairly general programs. Scores on the Piaget test were clearly not predictors of generality of programming style.

What about attitude? Perhaps students were more successful in writing Logo programs because they had a positive attitude toward the course. While the average attitude scores for each of the three attitude assessments tended to be positive, overall the average attitude scores decreased over the period of time that Logo was taught. Further, students with neutral or even slightly negative attitude scores were able to produce respectable programs. Neither more positive nor more negative attitudes seemed to relate to programming ability.

Examination of attitude scores apart from other data showed that there was a significant difference between the scores of males and females on the first attitude test and no significant difference between males and females on the third test. The attitude scores of males fell; while the scores of females increased slightly. Does this imply the the girls were affected differently than boys by the course? One hypothesis might be that the attitudes of the boys was initially inflated and became more realistic as the course progressed.

This attitude difference between males and females caused me to examine the question of success from another point of view. Perhaps, in addition to thinking about individual successful students, we should ask whether a course is successful. Perhaps a Logo course is successful if a student can write Logo programs of any sort. Perhaps it is successful if the student leaves the course with a realistic attitude toward computing.

As is often the case with Logo, this study seems to raise more questions than it answers. Those of us who teach Logo see exciting things happening to our students, but they often defy measurement. In the midst of our day to day teaching, it is easy to get caught up in the rigidity of the school system and content ourselves with collecting homework and assigning grades. Perhaps we need to think about what is meant by success, in both our students and our courses before we next enter the classroom. Perhaps we need to find a way to make the grades assigned match more closely the things that we philosophically consider to measure success with Logo.

[1] Hudson, Tate Mason Brewster. A Study of Documents to Measure Early Adolescents, Their Piagetian Stages of Development, IQ Levels, and the Interaction of These and Other Variables in Predicting Success on a Grammar Task. Dissertation, University of Akron, 1983.

[2] Burrowes, Sharon. A Study of the Effects of Using CAI to Teach Signed Numbers to Seventh Grade Gifted Students. Dissertation, University of Akron, 1983.

N. Livesay and C.F.Nodine
Department of Educational Psychology
Temple University
Philadelphia, Pa. 19122-004

A PARENT'S PERSPECTIVE ON LOGO: THEORY, RESEARCH, AND EDUCATIONAL PRACTICE

Much as our culture attempts to idealize our remembrances of our"golden" schooldays, who among us can look back without clear, powerful memories of pain, suffering, and humiliation? And who among us would not long to spare our children such trauma? No one is more desparately convinced of the need for radical educational change than parents who have watched animated, radiantly hopeful first-grade faces harden into the dulled, numbed expressions characteristic of fourth-graders.

The promise of discovery or Piagetian learning is particularly appealing for parents who, indeed, recognize it as the child's "natural," seemingly effortless mode of learning. Discoveries depend upon the child's own questions, not those of the teacher; thus they cannot be forced, and, therefore, they cannot be guaranteed. For educational traditionalists (including, almost by definition, most educational professionals), the weakness of discovery learning is its very lack of predictability and "efficiency." According to Perkins (1985), the learning opportunities which unquestionably exist within the Logo environment often simply "do not get taken."

Computers in the School: Facing the Realities

In the elementary school where we, in the role of parent-volunteer, have been introducing children to Logo for the past year, there is perhaps one computer for each 80 students, and groups of 3-5 children have access to the computer for about 1/2 hour per week per group. Many teachers admit to being indifferent to or frightened by either computers or the prospect of learning something new (!!) This is so pathetically far from the ideal situation for learning Logo that we have wondered whether our efforts are of any value at all. Yet these children confide that they are amazed to have found a use for school math, they animatedly discuss the size of an angle modification on a group project, they brainstorm. Parents, unknowledgable about computers, who happen to see these children working say, "This is what I want my child to do." As is commonly reported in Logo environments, these children beg to be allowed to work on their projects at recess. Children beg to learn. Opportunities do get taken, after all!

Active vs Passive Learning

The confrontation between educational traditionalists and revolutionaries is representative of the type Piaget called "dialogues of the deaf," (Piaget, 1972) in that the argument reduces to fundamental and possibly irreconcilable (Reese & Overton,1970) assumptions about the nature of knowledge and learning. For those who believe in the "active organism," learning must necessarily be self-induced, and motivation becomes, quite automatically, a critical determinant of what will be learned. For those who subscribe to the model of the "passive organism," however, opportunities for learning will only be taken if the environment is appropriately "engineered" to trigger such a reaction (Perkins, 1985) , and "motivation," if it can be presumed to exist at all, is suspected of potentially distracting the learner from the learning task (Lepper, 1985).

This confrontation of theoretical assumptions separates American Logo researchers into two camps: the pro-Logo, "active organism/ motivation-is-important" camp and the con-Logo, "passive organism/ motivation-is-irrelevant-or-possibly-harmful" camp. Research for both camps has focussed on the issue of cognitive gains. Research design reflects the fundamental ideological and paradigmatic differences between the two camps and, furthermore, suffers from a disregard for power (the probability of finding a significant difference when one does in fact exist). Thus the conclusions, positive or negative, are in either case often foregone.

Revolution vs Evolution

British Logo advocates, who have from the start promoted "evolution" rather than "revolution" (Ross & Howe, 1984), have recently begun to distance themselves from global cognitive concerns and are focussing on "more modest but very positive" (McShane & Simon, 1985) gains.

Evolution is after, all a fact of life, while revolutions can easily go awry. We all accommodate as well as assimilate , we all make compromises: we try to make them thoughtfully, without compromising our visions. And so, as parents and educators with visions, ideas and ideals, we preach revolution though we must practice evolution. Here, in our less than ideal situations, we must not let the brightness of our ideals blind us to smaller signs of real change.

References

Lepper, M.R. "Microcomputers in Education: Motivational and Social Issues." American Psychologist: Jan. 1985, pp.1-18.

McShane, J., & Simon, T. "Hard Facts and Hype." The Times Educational Supplement: Oct.25, 1985, p.38+.

Perkins, D.N. "The Fingertip Effect: How Information-Processing Shapes Thinking." Educational Researcher, August/September 1985, pp. 11-17.

Piaget, J. Insights and Illusions of Philosophy. London: Routledge & Kegan Paul, 1972.

Reese, H.W. & Overton, W.F. "Models of Development and Theories of Development." In Goulet & Baltes (eds) Life-Span Developmental Psychology: Research and Theory. New York: Academic Press, 1970.

Ross, P., & Howe, J. "The Design of Edinburgh Logo." University of Edinburgh DAI Research Paper #232, 1984.

Terrapin™
The Logo People
Terrapin, Inc., 222 Third St.
Cambridge, MA 02142 (617) 492-8816

Playing With Words:
A Bridge Between Turtle Graphics and List Processing

Alison Birch
Terrapin, Inc.
222 Third Street
Cambridge, MA 02142

It's okay to be a beginner in "words and lists."

There are two seemingly separate parts of Logo–turtle graphics and list processing. To build a bridge between them implies that there exists a gap that needs crossing, and people can argue (as they frequently have) that this is true.

In fact, the bridge between turtle graphics and list processing is the wrong bridge. A more interesting and appropriate bridge is one that leads beginners in "words and lists" to more advanced list processing.

The much used phrase "no threshhold, no ceiling" does, in fact, refer to both aspects of Logo–turtle geometry and list processing. It's just that turtle graphics is usually taught with too low a ceiling, and list processing is taught with too high a threshhold. The Logo philosophy, which is really an educational philosophy, emphasizes exploration and discovery. This philosophy is usually followed in turtle graphics, but not in list processing. However, through projects that encourage play with language, you can enjoy a similar feeling of exploration and discovery that you were able to experience when you first started to move the turtle around on the screen.

Programs will be presented that were developed with and used by 4th - 11th grade students at The Phoenix School in Cambridge, MA.

Several topics will be discussed including

- using tool procedures
- recursion
- programming style
- interaction
- data structures

Procedure listings will be available in a handout.

Data Representations in Real Logo Projects

Margaret Minsky and Cynthia Solomon

We will discuss different representations of data in Logo project examples taken from our book, Logo Works. There are a variety of strategies used in these projects to represent the data that programs use, for example, storing data in list structure, storing data in variables, representing programs as data. These representations overlap. A representation may look like all of these at once; you decide which to call it depending on how you are thinking about it at the moment. In discussing projects you can vary your selection, but when you are writing a book, you have to decide what to call it at the time you write.

We have many voices in our book; the voices of the authors of the projects and our voices writing about their projects and editing their writing. We had to decide in each project write up how to talk about the data representations used, and we had to develop a more-or-less consistent language for this. Sometimes the representation issues are discussed explicitly and sometimes they are left implicit.

In this talk we would like to make the representation contrasts between projects more explicit and detailed than in the book, as well as discussing the decisions made in how to write about them. We would like to contrast the representations in several programs including a simple sentence generator, an interactive dialog program, a diagram generator, and a game program. We will discuss both the representation the choices that Logo project creators made in their programming, and our choices in how to describe these data representation categories.

THE ISSUE OF INSTANT LOGO

Paula Cochran and Glen Bull
Curry School of Education
University of Virginia
Charlottesville VA

There are many ways to introduce Logo to young children. The difficulty of ensuring that it becomes empowering for them is a recurring topic in teacher training courses and in the Logo literature. Problems frequently reported by teachers include:

(1) the regular Logo commands require too much typing for a young child

(2) the use of large, double-digit numbers does not match the child's developmental level

A form of Logo which is often called "Instant Logo" is one of the more popular solutions to these problems. In most Instant Logo programs (sometimes called "Easy Logo" or "Single-Key Logo"), the child only has to press a single key to make something happen on the screen. For example, the child might press F, causing the turtle to immediately move forward 10 steps. Writing a short Instant program is not overly difficult, and, in fact, is a common exercise in beginning Logo texts.

The Issue of Instant

One of the most highly touted characteristics of Logo is that it permits playful exploration of numbers. However, in traditional versions of Instant Logo, the child does not have an opportunity to input a number at all. This is just one of the limitations which has led critics to note that most versions of Instant take the power of Logo out of the child's hands (Papert, 1984; Tipps, 1983).

This is true in many cases. Fortunately, an Instant Logo doesn't have to be so limited. Our experience with looking at ways in which Logo can complement a pre-academic or early academic curriculum have lead us to require certain features in Instant programs we use.

Guideline 1

A written message on the screen should tell the child what is happening after each keypress. For example, when the child presses the letter F, we would like the word "FORWARD" to appear on the screen (the whole word, not the abbreviation). The screen message doesn't have to match the equivalent regular Logo command. Occasionally, for example, we have substituted the word WALK for FORWARD when working with cognitively delayed children.

Guideline 2

The child should have the opportunity to supply numeric inputs to procedures. This can be a straightforward modification.

```
TO F
   (TYPE [FORWARD] SPACE)
   MAKE "NUMBER INPUT.NUMBER
   FD :NUMBER
END
```

This version of our F command allows the child to type in a number. It also prints out the full word FORWARD, followed by a space. Since the child may want to type more than a single digit (such as 10), the program waits for a carriage return before executing the Instant command.

Guideline 3

An Instant program should provide error messages that are meaningful to a child. In regular Logo, if you type FORWARD and press RETURN without entering a number, Logo is apt to present an error message such as:

```
FORWARD NEEDS MORE INPUTS or

LOGO DOESN'T LIKE [] AS INPUT TO FORWARD
```

Adults may recognize that Logo is telling them that by pressing RETURN they entered an empty list as input, but this error message isn't helpful for a five-year-old child.
It also isn't helpful if the program crashes when the user makes a slight mistake. We anticipate that there are two types of errors young users of Instant will make:

(1) They will press RETURN without entering a number.
(2) They will type a letter instead of a number.

The procedure which inputs a number should check for both of these possibilities, and provide a suitable error message. The error message should be customized by the teacher. For example, if "Please type a number" is not appropriate, it should be changed to any suitable message.

Guideline 4

An Instant program should encourage use of numbers which are meaningful to the child. For a child who is just becoming familiar with small numbers even the Instant described so far can be a bit frustrating. If she uses the numbers 1 to 10 as inputs, for example, not much is going to happen on the screen. In fact, small turns are frequently not visible at all in regular Logo.

The dilemma here is that a pre-set number is extremely limiting, but use of the small numbers with which the child is familiar is unrewarding. Solution: make the small numbers visible, and thus more interesting. This involves multiplying the number typed by the child. We might decide that for every 1 step requested, the turtle moves 10 turtle steps, for a ratio of 1 to 10.

```
TO F
   (TYPE [FORWARD] SPACE)
   MAKE "NUMBER INPUT.NUMBER
   FD :NUMBER * 10
END
```

The turn commands will need some adjusting, too. We found out the hard way, when one mystified 6 year old typed F(orward) 5 (which moved the turtle forward 50 turtle steps, in 5 easy-to-see jumps) and then R(ight) 5 (which barely turned him at all). The same range of numbers should be effective as input to distance and direction commands. We generally multiply turn inputs by 10, so that an input of 3 results in a 30 degree turn, etc..

Corrollary to Guideline 4

A young child should be able to see each step the turtle takes (and count along). On most computers, when FORWARD 50 is entered, it looks as though the turtle took one giant step instantaneously. The turtle should pause in between steps. Then the child will be able to count along as the turtle moves. We often add a WAIT procedure, causing the turtle to stop briefly after each step.

Summary

Use of Instant Logo can be empowering for even young children. Instant can be used to facilitate a wide range of instructional objectives. We suggest that an effective Instant Logo program should have the following characteristics:

1. capability for entry of any number
2. appropriate turtle step length
3. appropriate turtle step speed
4. visible results with small numbers (for young children)
5. messages which tell what is happening on the screen
6. meaningful error messages
7. easy exit and resistance to crashing

References

Papert, S. (November-1984). Misconceptions about Logo, Creative Computing, 10 (11), pp. 229, 232.

Tipps, S (December-1983). The issue of instant (Tipps for Teachers), National Logo Exchange, pp. 6-10.

Charlotte L. Scherer, Ph.D.
Director of Clinical and Computer Labs
College of Education
Bowling Green State University
Bowling Green, Ohio 43403-0243

Early Childhood Majors Teach Logo to Preschoolers

Introduction

A pilot project during the spring of 1985 was followed up by two semesters in which early childhood majors at Bowling Green State University introduced four and five year olds to the computer by using Logo and other programs appropriate for preschoolers. The pilot project had two major purposes: 1) to try out Logo and other preschool packages on some five year old nursery school children, and 2) to help this author and her staff figure out just what these children would and could do with computers. This pilot project is fully described in the Proceedings of the 27th Annual International Conference of the Association for the Development of Computer-Based Instructional Systems (Scherer, 1986).

Since the pilot was a success, we were ready to tackle a more complex project in the fall. The goals for this project were threefold: 1) teaching pre-service early childhood majors: a) computer awareness, b) the rudiments of single keystroke Logo and how to use several commercial programs suitable for preschoolers, c) methods of working with preschoolers using computers; 2) introducing young children to the computer by: a) teaching a single keystroke version of Logo commands, b) teaching games and programs that would reinforce letter and number concepts, c) using a wide variety of off-computer activities to reinforce the on-computer lessons; and 3) discovering: a)whether young children can make significant progress with the concepts of right, left, directionality, shapes and angles by using Logo, b) whether the computer can aid preschoolers in learning numbers and letters, c) whether the computer can help young children learn the rudiments of problem-solving approaches.

Project Activities and Methodology

This project involved first training a group of early childhood majors and then overseeing and complementing the work they did to teach the preschoolers how to use, enjoy and learn from the computer. Thirteen early childhood majors in a sophomore level preschool methods class were introduced to the computer in the fall and taught the rudiments of single keystroke Terrapin Logo. They were also shown several other preschool packages, namely, Sticky Bear ABC's, Numbers and Shapes (Optimum Resource, Inc., 1984), MECC Pre-reading/Counting disk (MECC, 1981), Muppet Learning Keys (Sunburst, 1984) and Teddy's Playground (Sunburst, 1985). Meanwhile, arrangements were made with the nursery school for 18 four year old children (in groups of nine) to participate in the project. Four year olds were chosen primarily because the instructor of the early childhood class preferred that the students should get experience with children younger than Kindergarten age.

The children came for 45 minute sessions twice a week for five weeks. Each early childhood major was assigned one or two children with whom to work. The 45 minute sessions were divided into three parts, with the greatest part given to off computer activities. It is firmly believed that the only reasonable way to introduce such young children to computers is to recognize the importance of providing a variety of experiences, including physical activities, and to keep in mind the short attention span of this age group. Thus, there was a teaching period, a period of physical activity which reinforced some of the commands and concepts introduced during the teaching period, and a period of time for using the computers.

Detailed lesson plans were produced for each of the sessions by the Project Director. These plans helped the early childhood majors and other staff assistants to know the objectives for each session. They also were a ready reference for future planning and ensured that a logical sequence of events was followed for the children's learning and skill development. The plans were written on the computer so they could easily be revised.

Logo was chosen as the main activity because of its reputation for putting children in command of the computer. Logo 's color turtle graphics readily lend themselves to use by young children who can make designs, shapes and "pictures" using a single keystroke version of Logo (Abelson, 1982; Papert, 1980; Rieber, 1985; Watt, 1982).

It is important for very young children to have many concrete experiences, then move from these concrete experiences to a somewhat abstract level, and finally to move to the much more abstract level of a computer screen. Hence, the project included in the off-computer sessions, many activities with physical objects or using the children's bodily movements. These off-computer sessions were at first directed by this author or one of the Lab staff. Later, some of the university students took the responsibility for planning and implementing these activities. Some typical activities were having the children move like turtles with their bodies, then having them move turtle objects; next they would program a Big Trak toy tank (Milton Bradley, 1979) and a Turtle Tot floor turtle (Harvard Associates, Inc., 1983, 1984) and finally they would work on the computer itself with the screen triangle that is the Logo turtle. Following such a sequence allows for a great deal of initial and immediate success, with little or no frustration. It was fairly easy also to individualize and challenge those four year olds who were more mature and ready for certain concepts and to provide less challenging activities for those who needed more readiness time (Durkin, 1970; Papert, 1980).

Because it was felt that the children needed a change of pace and in order to reinforce the sight recognition of the letters, numbers and shapes, some other commercial programs for preschoolers were used to supplement the Logo activities. These programs helped the children become more familiar with the computer and helped with skills they were learning in nursery school.

The Second Semester

This fall project is to be followed up by yet another project, using many of the same early childhood majors to work with five year olds who can go beyond the single keystroke version of Logo. In this way, the early childhood majors will capitalize on what they already know and build new skills, while a second, but older group of nursery school children will be introduced to the computer and will move into slightly more advanced work with Logo. Slides have been and will continue to be taken of the projects. These will be shown and a summary of the results of the two semester projects will be discussed in the proposed presentation. Both the nursery school children and the university students had positive reactions to the project and most of the goals of the fall project were effectively met.

Teachers and the Logo Computer Culture

Cynthia Solomon
80 Ellery Street
Cambridge, Ma. 02138

In this presentation, I would like to explore with you what roles teachers might play in computers and education. This discussion draws heavily upon a recently completed study, Computer Environments for Children: Reflections on Theories of Learning and Education (MIT Press, 1986), in which I focus on two images of computers in education--the computer as an interactive textbook and as an expressive medium. Whether the computer is acting as an interactive textbook or as an expressive medium, the teacher's role varies in different situations. A general issue emerges as to how teachers might learn their practice so as to integrate the computer's potential into their classroom activities. Here I put forth a view that presents one possible route into computing for educators. This path is based on my personal experiences in working with children and teachers. This path has a long history, starting over twenty years ago when I began my collaboration with Seymour Papert.

In my view the computer is an intellectual agent, operating in a culture and reflecting ideas of the culture. Logo, as a programming language, is used as the main means of communication with the computer. But what really guides this culture is the belief that naive as well as sophisticated people should be given access to powerful ideas in a dynamic and concrete way; computers can offer easy access to such ideas as well as models for how to build on them. Furthermore, although there is an accepted and widely held attitude that experience in programming computers is in itself stimulating, enriching, motivating, and valuable in many ways for children, high school students, indeed, everyone, there is a belief that more can be offered. The computer becomes a medium for self-expression and an instrument for one's own intellectual development. This process involves people, machines, and ideas. Each is linked to the others.

Perhaps the major question to be discussed is, How do teachers acquire knowledge about computers that they can put into practice in their daily teaching? Most courses on computers in education take a skills approach, whereby within a semester they provide the students with some of the syntax of a programming language or familiarity with word processing and assorted educational games. In contrast, I talk about what you might do while you take that course or the next one. That is, how you can take advantage of the computer's potential.

For many teachers learning about computers can be frightening. The machine, the keyboard, the floppy diskettes, the typing--All contribute to a kind of bewilderment and tension. I would like to be there to tell you to relax, it's okay. I would like to continue to give you the same advice I give children. I would like to be there to set your computer up so that you are not confronted by the bewildering array of machine parts and wires. But, I can say only that in the future computers will contain a library of information as well as space for personal work. There will be a significant breakthrough in speech recognition so that the computer will understand verbal commands as well as those passed to it from a keyboard. Of course, talking out loud can be a distraction in a classroom. Anyway, that is in the future, and the frustrations of communicating with the computer through a keyboard are momentary and fade away as you gain familiarity.

I emphasize first encounters because first encounters often leave lasting impressions. I want to help make that first experience a positive, almost magical one. Rather than concentrating on the syntax of a language or even a game I want to concentrate on the computer experience itself. I want to provide you with insight into the computer's potential as a powerful personal intellectual agent. I want, therefore, for you to be able to do something interesting that you could not do before using a computer.

Turtle geometry offers a multitude of possibilities to meet these criteria. Although there are other programming domains from which to draw, turtle geometry stands out as the most concrete and obvious. After issuing a few commands to a turtle on a TV screen, you have created a square, a star, a squiggly path showing the turtle's reactions to your commands. You are strongly impressed by the first twenty minutes of your computer experience.

This is the beginning of a remarkable intellectual journey toward mathland and the world of computational ideas. By going from the concrete to the abstract and from the abstract to the concrete, you develop procedures to describe particular actions by studying individual instances of that general behavior. For example, seeing that the process by which the turtle walks in a square path is the same as walking in a triangular path leads to an understanding of the total turtle trip theorem and the POLY procedure.

Computers offer a new opportunity to help teachers to enhance their teaching and understanding of children and to keep schooling from becoming an alienating experience. Whether or not this will happen is unclear. To make it happen, action needs to be taken now to reeducate educators to develop models of what might be possible.

Teacher Preparation

In discussing teacher preparation people tend to focus on what software packages to talk about and whether a teacher should be familiar with Logo, BASIC, or Pascal. In discussions about using computers in schools people tend to extend their concerns to whether computers should be in a laboratory or in individual classrooms. Rarely, do discussions focus on what kinds of learning environments might be developed to introduce educators to a computer culture. Furthermore, the amount of time set aside for teachers to become comfortable and knowledgeable is short. The resulting tendency is to encourage the popular beliefs that (1) educators in general cannot learn to program, and (2) programming as we know it today will disappear in a few years, so why learn about it now? Of course, educators can learn to program as easily as ten year-olds or they might find it impossible, depending on the kind of programming projects they are engaged in. As for the second objection, it is likely that programming techniques will change, but important ideas in computer science, such as naming, procedurizing, debugging, heuristic methods, and simulations, will not disappear. What is likely to change is the syntax and power of particular programming languages and the domains over which they operate.

Developing programming projects and debugging them can be a rich intellectual experience for teachers and children. In what follows I would like to explore in some depth some ideas about preparing people to work with children and computers. I want to situate this discussion in terms of the question, What does an elementary school computer teacher need to know? I am going to talk about my own experiences. In discussing this question, my strategy is to examine carefully the knowledge I bring to bear on teaching children to program. This knowledge, much of it tacit and intuitive, developed over many years; I now try to formulate it explicitly. As a teacher, I see much of my own development as acquiring (1) a repertoire of programming projects that make the power of programming techniques and concepts apparent to beginners, (2) a vocabulary for talking about structured programming, and (3) a sensitivity to the kinds of resistance that keep many adults and children from experimenting with mathematical ideas.

Out of experiences in this culture a new breed of teacher emerges: This teacher is thoroughly imbued with a coherent computer culture and its language. She knows how to use this language to talk interestingly about things people from outside the culture know and care about. This teacher has a fluent mastery of certain powerful ideas. She is thoroughly familiar with project terrains through which she will guide those who come for "instruction" (but will be given something better!). She has been there often! She knows how to observe people engaged in thinking, learning, puzzling, agonizing, rejoicing .... She knows (and can only know this through experience) when to intervene and when to let the learner struggle. She believes that the key goal for any learner is to improve his image of himself as a learner, as an active intellectual agent.

Teaching Teachers: Curriculum-Based Logo Projects

Marianne Handler and Sandra Turner
National College of Education

At National College of Education a two-quarter sequence in Logo is offered as part of a graduate degree program in Computer Education. In the three years we have been teaching Logo to teachers, the course content has gradually changed. Some of these changes have been based on our own evaluation of the course, while others have been influenced by the shifting focus in the field of computer education from programming to integrating the computer into the curriculum. The purpose of this presentation is to describe how our course for teachers has evolved so that it includes more emphasis on integrating Logo into the curriculum.

Overall Course Goals

All the students in our Logo courses are experienced K-12 teachers, so they come to the Logo course wearing two hats, that of the learner of a new language and its underlying philosophy and that of the teacher who wants to apply Logo to the classroom. Thus, while programming is a major component of the course, there is an equal emphasis on teaching methods and materials, creating a Logo environment, and using Logo to teach other curricular objectives.

Curriculum Projects

For the last three years each student in the first course in the sequence has completed two programming projects, one involving turtlegraphics and the other an interactive program involving simple list processing. A year ago, as a result of evaluating the course, we decided to add a curriculum-based project to the course requirements.

For this project the students wear their teacher hat. They design and develop materials that either help to teach a Logo concept, such as file management, or that use Logo to teach concepts in other curricular areas, such as mapping skills in social studies. The curriculum project need not involve programming, although in many cases the materials include both a Logo program and accompanying print materials. It must, however, be a finished lesson. Each lesson is demonstrated to the rest of the class and the materials are duplicated and shared. Through the sharing of projects, students acquire many new ideas for teaching Logo and for incorporating it into the curriculum. In addition, they finish the course with a set of materials that they can redesign for their own students' grade level.

In the second course in the Logo sequence students learn advanced programming concepts, such as recursive list processing, and apply these concepts in developing a program that they can use in their classroom. Since the class discusses the characteristics of a microworld and sees several examples, many students incorporate these characteristics into their own projects. Again, as students share their projects with each other, they see a variety of ways that Logo can be used in the classroom and they gain an appreciation of the power of Logo as a programming language.

Evaluation Criteria

In order to meet the college's requirement that grades be given and yet attempt to model a Logo environment, we developed evaluation criteria for both the programming projects and the curriculum-based project. The criteria for the programming projects include:

*error-free execution
*well-structured programming

appropriate use of subprocedures
meaningful variable names
minimum use of global variables
*well-formatted screen display
no split words
no scrolling
correct spelling and grammar
*user-friendly
*clean workspace

Criteria for the curriculum-based project include:

*clearly stated objectives
*instructional value
*appropriate for intended audience

With specific criteria in advance, students have a clearer understanding of what is expected, and our task of evaluation is easier and more objective.

Conclusion

Our Logo course for teachers addresses two of the concerns aired at Logo '85: the need for teachers to be exposed to a variety of examples of integrating Logo into the curriculum, and the need to give grades in a graduate-level course while trying to provide a model of the Logo environment. The curriculum-based projects we have incorporated into our course take the teacher learning Logo beyond using the language for its own sake to consider how Logo can be applied in the classroom.

In our presentation we will demonstrate examples of curriculum projects developed by our students and share the criteria that we developed for evaluating their work.

LEGO/LOGO

Stephen Ocko and Mitchel Resnick
February 24, 1986

INTRODUCTION

Lego/Logo is a computer-based system that allows children to combine the worlds of Lego and Logo. Using the Lego/Logo system, children can build machines with Lego building blocks, then write Logo computer programs to control the machines that they have built.

In this talk, we would like to discuss (1) our experiences using Lego/Logo in elementary-school classrooms and (2) our plans for building a curriculum around Lego/Logo activities.

OVERVIEW

The Lego/Logo system is being developed jointly by Interlego (of Denmark) and Microworlds Learning Inc. (of Cambridge, MA). The system is based on a special interface box that Interlego plans to begin marketing in 1987. Through this interface box, students can send signals to Lego motors, and receive information from Lego sensors (such as touch sensors and light sensors).

To program their machines, students use a specialized version of Logo. This modified Logo includes primitives like ON, OFF, RD (for reverse direction), and SENSOR, so that children can program their machines using natural, intuitive language.

Lego/Logo is an interdisciplinary activity, involving important ideas from mathematics, engineering, architecture, and science. While working on Lego/Logo projects, students experience what it is like to actually work as a designer, inventor, and engineer. They create new ideas, build prototypes, test them out, and modify them to improve performance.

Children have already used Lego/Logo in a wide variety of projects. One fourth-grade girl, for instance, built an automatic door which opens whenever a touch sensor is triggered. A fifth-grade girl built an entire "chocolate-carob factory," complete with computer-controlled conveyor belts. Other children have designed vehicles that automatically change direction whenever they bump into walls.

The Lego/Logo system is currently being tested at the Hennigan school in Boston. Between April and June, the system will be tested at other schools in Boston, New York, and Connceticut.

OBSERVATIONS

Through our testing at the Hennigan school, we are gaining a better understanding of the potential applications of Lego/Logo. Among our observations:

(1) Learning key concepts. We have found that many students find it easier to learn certain mathematical concepts (such as ratios) in the context of Lego/Logo activities as opposed to traditional classroom approaches. Many students also seem to learn certain programming concepts (such as conditionals) more easily in the Lego/Logo environment. In addition, students learn certain engineering concepts (such as feedback) that traditionally have not been addressed in the elementary-school curriculum.

(2) The affective side of Lego/Logo. We believe that one reason student learn new concepts so quickly in the Lego/Logo environment is that they care so deeply about the things that they are making. We have found that children show an incredible attention span and level of involvement while working on Lego/Logo projects.

(3) Lego/Logo as a confidence-building activity. For reasons that we are studying but do not yet fully understand, many children who have difficulty with "technical" subject matters (such as math and science and even computing) take easily and enthusiastically to the Lego/Logo combination. The confidence that they gain in Lego/Logo is then transferred to other activities. This aspect of Lego/Logo could be especially important to "special-needs" children. Many students who are termed "learning disabled" have strong spatial skills that might make them comfortable and talented in the Lego/Logo environment.

(4) Gender differences. We have been pleased to discover that the Lego/Logo environment is as appealing to girls as it is to boys. According to "conventional wisdom," both Lego and computers are intimidating to girls. But we have found that the girls have enjoyed working with Lego/Logo and have built some of the most interesting projects.

(5) Students as scientists and inventors. Lego/Logo provides an environment in which students want to experiment and explore. We have observed many students constructing theories and testing them out. (Will a car go faster if it is heavy or light? Will it go faster if it has one motor or two?) We believe that Lego/Logo activities have enabled many students to view themselves as inventors (of theories and of machines) for the first time.

(6) Social interactions. Many activities in the Lego/Logo room are group activities. Students often work together as a team in their projects, sometimes one student focusing on the Lego and another on the Logo. As a result, students learn the value of sharing and cooperation. Teachers have

commented that social dynamics of their classes were much improved after working in Lego/Logo activities.

CURRICULUM

We have designed a series of activities to guide the use of Lego/Logo in the classroom. Our goal is to introduce students to the key concepts of Lego/Logo, then allow them to experiment and explore on their own.

By working on these projects, we believe that students gain important insights into many concepts that are typically convered in the elementary math and science curriculum--such as measurement and computation. In addition, they can explore some concepts that are traditionally taught only at higher grade levels.

We begin with a non-computer activity. We tell each student to build a (non-motorized) Lego car, and race the cars down a ramp. Students measure how far their cars go, then modify the cars to try to make them go further. Students keep records of their ideas and results in their "Inventor's Notebook."

The computer is first introduced as a tool for timing the cars as they go down the ramp. Then, students begin adding motors to their cars and writing programs to control the motors. At this point, we encourage students to think about other (non-car) projects. Students have built everything from an amusement park roller coaster to a computer-controlled pop-up toaster.

At this point, we also encourage students to build and program a "Lego turtle." Students can equip the turtle with touch sensors so that they can detect walls. Or they can equip the turtle with a light sensor, so it can detect black lines on a white tabletop (or white lines on a black tabletop). We believe that these projects form a good foundation for classroom discussion of animal behavior and artificial intelligence.

At present, we are focusing our curriculum efforts on upper-elementary grades. Eventually, though, we plan to push the Lego/Logo curriculum in both directions. In the long term, we believe that Lego/Logo can be used in meaningful ways all the way from kindergarten to high school (and maybe beyond).

The Logo - PostScript Connection

Scott R. Garrigan
Educational Technology Program
College of Education
Lehigh University
Bethlehem, PA 18015

PostScript is a page description language that is now widely available on laser printers. The author has explored several features of *PostScript* using an Apple //e microcomputer and an Apple LaserWriter printer the Educational Technology Program at Lehigh University's College of Education. The presentation will describe the features that make *PostScript* of interest to the Logo community, particularly the Logo publishing community. Like Logo, *PostScript* is an powerful extensible language with a rich graphics capability. Each language can handle both turtle geometry and coordinate geometry, and each one supports powerful features such as procedures and recursion. Many Logo concepts can therefore be implemented in *PostScript*, and Logo graphics can be superbly printed through *PostScript*. The presentor will demonstate a number of practical applications, developed at Lehigh, in which the new language can help the Logo community:

1. Camera-ready turtlegraphics may be printed that are of publication quality.

2. Some Logo graphics concepts may be more clearly demonstrated at ultra higb resolution, far surpassing the resolutions of computer monitors and most printing devices.

3. Logo procedures and graphics may be used to produce artistic creations (such as mandalas, or artificial landscapes using fractal geometry) with finer detail than is possible with monitors and most printing devices.

4. Turtlegraphics may be combined with publication quality text in an attractive variety of sizes and fonts for camera-ready copy.

A comparison of *PostScript* to Logo as formal programming languages can provide insight into the relative merits and features of each language. For example, LISP and Logo are list-oriented languages; *PostScript* and Prolog are object-oriented languages. Several open-ended questions will be introduced and addressed in the discussion. Are there educational applications of *PostScript* that complement the educational applications of Logo? Can *PostScript* be as versatile as Logo? Can interpreters he easily designed to convert Logo to *PostScript*?

Practical Benefits to the Logo Community

Logo users can use *PostScript* to prepare their graphic concepts for publication more easily than ever before. *PostScript* supports the rotational and translational transformations that make turtlegraphics possible. Since *PostScript* allows printing at the maximum resolution of the printer, and *PostScript* printers have resolutions from 300 dpi (dots per inch) to 2500 dpi, turtlegraphics can he printed at true publication quality.

For example, the Apple // high resolution graphics screen is a bitmap containing 278 pixels horizontally and 192 pixels vertically, with a 53,376 bits in the matrix. A normal screen dump of such a Logo graphic on a common dot matrix printer would yield a resolution no greater than 278 x 192, regardless of the resolution of the printer or the scale at which the graphic is printed. In constrast, the same graphic could be printed with *PostScript* from the Apple // on an Apple LaserWriter printer at a resolution of 300 dpi, or a page resolution of 2400 x 3500 pixels (more than 8 megabits). Through the use of higher resolution laser printers, such as the *Lintronic* series which support *PostScript*, resolutions of 1000 dpi to 2500 dpi are possible (surpassing half a gigabit per page)!

Many turtlegraphic concepts are more clearly demonstrated at greater resolutions than are possible on the screens of today's microcomputers. For example, only a few levels of fractal geometry can be displayed with the resolution of the Apple // or Macintosh screen. If the graphic extends beyond the screen boundaries, we can only see a window onto the larger pattern. We are limited both by depth and bounds in fractal display. The Mandelbrot *C* or *Dragon* curve (McDermott, 1983) is an example of a fractal shape that cannot satisfactorily be displayed on a common Logo monitor. *PostScript* cannot only display a single page at very bigh resolution, but it can also break larger patterns into several pages (up to its internal memory and stack limitations).

Logo has the capability of producing magnificent geometric creations, but the display devices and computer memories available to Logo users have generally been a limiting factor in the complexity of Logo grapbics. With *PostScript*, Logo users can pursue artistic creations that would not be displayable with normally available computer systems. Examples of single and multi-page turtlegraphic art created at Lehigh will be presented.

Many conference attendees publish articles on Logo. With *PostScript*, graphics (including halftones) can be combined in any way with text to produce camera-ready, publication quality copy. The presentation handouts will be examples of these techniques.

What is *PostScript*?

The similarities and differences between Logo and *PostScript* will be demonstrated. Formally, *PostScript* is a device-independent, page description language. It is used in laser printers, such as the popular Apple LaserWriter, to allow the printer to create text and graphics to the maximum resolution of the printer, rather than to the resolution of the computer screen. *PostScript* is based upon the Forth language, and therefore is a threaded, interpreted, extensible, stack-oriented language which supports procedures and recursion, and uses postfix notation. It is a full-featured language with about 250 operands (or commands), only one third of which are related to graphics (Adobe, 1985). *PostScript* offers many features not available in most Logo implementations, such as matrix manipulation, smooth scaling, and screen fills. *PostScript* also offers operators similar to LISP's MAPCAR.

PostScript is generally used by an application program such as *Microsoft Word* on the Apple Macintosh computer. The application program postprocesses its output into a text file containing a subset of *PostScript*. The program then sends the *PostScript* text file to the laser printer where it is processed by the *PostScript* interpreter. The *PostScript* language is completely device independent, and the output of the program is not converted to the device space of the printer hardware until the last moment, thus guaranteeing printing at the maximum resolution of the particular printer.

Although many people have used the Macintosh/LaserWriter combination, few people know that any computer can be used to create a *PostScript* program and send it to the printer. In fact, the computer can even be used interactively as a terminal, sending immediate commands to the *PostScript* interpreter in the printer and receiving responses from the interpreter on the computer monitor.

PostScript has several important similarities to Logo, the most important of which are the ability to do turtlegraphics, procedures, and recursion. Both languages can treat data as program and vice versa. But *PostScript* and Logo have some fundamental differences that may establish a limit to their ultimate compatibility. Logo treats data as linked atoms and lists; *PostScript* treats all data as objects. *PostScript* is stack-oriented and uses postfix notation, compared to the prefix notation used by Logo. Logo has its commands; *PostScript* has only operands which generally operate on objects at the top of a stack. The inputs to a *PostScript* procedure, for example, must be found on top of the stack prior to invoking the procedure.

A Logo-to-*PostScript* Translator

The three methods of constructing a Logo-to-*PostScript* translator that have been explored at Lehigh will be presented. If the program editor has a supervisory language such as AppleWriter's WPL, a mini-translator can easily be programmed in the supervisory language. tor can also be prognd contrasted as to their ease of use, effectiveness, and difficulty of implementation.

References

Adobe Systems, Inc. *PostScript Language: Tutorial and Cookbook.* Reading, MA:Addison-Wesley, 1985.

McDermott, J. Geometrical forms known as fractals find sense in chaos. *Scientific American*, December 1983, *14(9)*, 110-117.

COMMUNITY SCHOOL DISTRICT #3
300 West 96th Street
New York, New York 10025

April 22, 1986

Presentation Proposal For: The Third International Logo Conference
Massachusetts Institute of Technology
Cambridge, Massachusetts

Prepared By: Fred S. Goldberg
Director
Manhattan Technical Assistance Center
134 West 122nd Street
New York, New York 10027

Presentation Title: TeleLogo: Expanding Logo Environments Through Telecommunications

Background:

TeleLogo is a joint activity of the Manhattan Technical Assistance Center, New York City Board of Education and Community School District 3, New York City. The developers and principal facilitators of the project are Fred S. Goldberg, Director, Manhattan Technical Assistance Center, and Gwendolyn Brown, Director of Computer Education, Community School District 3. The students and teachers involved in the TeleLogo project come from the elementary component (grades 3-5) of the District 3 Computer School. Inquiries or other correspondence may be directed to Fred S. Goldberg.

Special Presentation Requirements:

- One Apple IIe microcomputer with 2 disk drives
- One Modem
- One outside phone line
- One large screen (RGB) color monitor

/gyh

TeleLogo

Project Description:

TeleLogo is an attempt to use telecommunications in general, and an instructionally based electronic bulletin board in particular, to enhance the Logo learning experience. Children from various schools in Manhattan, New York City, cooperate in creating Logo programs, sharing school activities and offering personal reactions while learning the concepts and mechanics of teleprocessing. The linchpin of the project is participation in the <Big Apple Bulletin>; that is, the <Big Apple Bulletin> of the New York City Board of Education serves as the electronic host and vehicle for this endeavor. The version of Logo used in this project is LogoWriter, an L.C.S.I. product.

Presentation - Part I:

With A Little Help From My Friends

In a "proper" Logo environment, children freely ask other children for help. Using a classroom or laboratory model, this usually means asking a child next to you or somewhere else in the room for assistance. In an environment expanded through telecommunications, the child next to you may be a child in the next school, district or city.

Utilizing the uploading and downloading capabilities of the <BAB>, this portion of the presentation will attempt to go online and illustrate the process behind, as well as the content of, bulletin board exchanges of Logo experiences. Teachers and/or children from Community School District 3 will be present to describe their activities. Activities may include:

- Logo projects
- Sharing programming "secrets"
- Literary descriptions of school programs using the word processing feature of LogoWriter

Presentation - Part II:

There's A Turtle In The Telephone!

Being engrossed in a learning experience makes the acquisition of knowledge and skills seem transparent. Knowledge and skills related to data transfer are taking an increasingly dominant role as technological "survival" skills. Through TeleLogo, these new skills may be acquired vis a vis a Logo bulletin board activity.

TeleLogo

One may say that a microworld sets the stage for learning some thing or some process while doing what is interesting. For the children participating in TeleLogo, the interesting involvement is in the Logo activity, however, telecommunications and concomitant data transfer skills become the hidden activity. This portion of the presentation involves teachers and/or students describing the telecommunications skills learned while engaged in the social activity of sharing Logo experiences. Telecommunication skills may include:

- Dialing and connecting to a remote computer
- Navigation of bulletin board menus
- Uploading and downloading text files and programs
- Running a downloaded Logo program

A Theoretical Framework for the Development of Metacognitive Abilities in Logo Environments

In his seminal work on the use of computers to develop cognition, Papert proposed that in Logo programming children will develop thinking skills through reflective, and reflexive, thought. This idea emanated from the dual influence of Piaget's constructivism and artificial intelliegence. One of the strengths of the artificial intelligence (AI) approach to studying people's information processing is the provision of a concrete embodiment for abstract cognitive processes. Papert proposed to teach AI to children so that they, too, will be able to think concretely about their own cognitive processes.

These broad roots have provided a general structural framework. However, like most frameworks, this one is not tightly organized enough to serve the needs of researchers and practitioners. Minsky and Papert's more specific society-of-mind theory generally has not been used as a basis for investigations and implementations of Logo environments.

A Componential Theory: Sternberg

Sternberg's theory may prove fecund. It shares some characteristics of the society-of-mind theory in that it too posits information components that communicate cooperatively with one another to solve problems. However, it has several additional advantages. For example, it contains specifically identified components and a hierarchical organization that facilitate its application and interpretation. Sternberg hypothesizes that different types of processes are carried out by separate components, elementary information processes that operate upon internal representations of objects. There are three kinds of components. Performance components are involved in the actual execution of a task. They perform such tasks as encoding, representing, and comparing information. Knowledge-acquisition components are processes used in gaining new knowledge and in creative thought. They *selectively* encode, combine, and compare information to determine what is relevant and to relate new information to old knowledge. Metacomponents are the "executives"; they control the operation of the system as a whole and are utilized in planning and evaluating all information processing. Metacomponents include: deciding on the nature of the problem, choosing a strategy for combining performance components, selecting a mental representation, allocating resources for problem solution, and monitoring solution processes. Metacomponents are the fundamental sources of intellectual development.

Use of Components within Logo Programming: Implicit Rationale

Children engaged in Logo projects are grappling significant, complex problems of their own design. This necessitates that they use each metacomponent at each phase of completing their projects. Thus, children's employment of metacomponents and knowledge-acquisition components might be frequent within a

Logo environment. The proposal that Logo programming will strengthen executive (i.e., metacomponential) and learning (knowledge-acquisition) skills has been made repeatedly. The theory proffered here is that as children construct and infer the consequences of causal sequences, they are engaged in the construction and modification of cognitive models and thus develop metacognitive and knowledge-acquisition abilities.

How do metacomponents develop? Sternberg (1985) has posited that cognitive development results to a large extent from the ability of the metacomponents to "learn from their own mistakes." They acquire knowledge about where, how, why, and especially when the various components might be best applied. The solution monitoring metacomponent not only keeps track of what has been done, what is currently being done, and what needs to be done, it also controls intercommunication and interactivation among the components, and thus has been termed a "metametacomponent." The question is, does Logo programming have the potential to facilitate metacomponential development, especially the crucial component of solution monitoring? Specific characteristics of (certain) Logo environments that appear to offer advantages for the development of each of the components will be discussed in the presentation. As one example, consider solution monitoring. Markman stated that the more actively people engage in inferential and constructive processing of information, the more likely they are to recognize information-processing failures. She suggested that children practice inferring consequences of causal sequences, finding problems, and enacting instructions. Logo programming involves operations of transforming incoming information in the context of constructing, coding, and modifying such causal sequences. Errors in turtle graphics projects are easily recognizable by children; that is, occurence of errors is highly visible. The nature of the actual "bug," and the steps to take to eliminate it, of course, are not as easily seen. However, Logo does provide aids for this activity in its graphic depiction of errors, explicit error messages, and easy to use editors. Thus, "debugging" programs in Logo that do not do quite what was intended provides children with valuable experience in utilizing their monitoring skills and provides children with a metaphor for understanding the process of solution monitoring in a variety of situations.

Consciousness of Components in Logo: Explicit Awareness Rationale

Children may receive substantial opportunities to engage in executive processing while programming in Logo. Note that while children may become explicitly aware of their errors and may develop stronger metacomponents, they do not have to be (and usually are not) cognizant of the existence of the metacomponents themselves or of their own metacomponential functioning. A unique claim is that Logo fosters explicit awareness of cognition. This provides another mechanism for the development of componential skills. It may be possible for children to learn simple notions about metacomponents, then use that knowledge to determine how to approach the solving of problems, and finally begin to use the knowledge automatically, without conscious direction. Their use of these processes--initially unconscious and ineffective--may become first conscious and more effective (though slow), and finally, unconscious and expert (automatic and fast). In other words, metacognitive experiences would provide declarative knowledge which is originally interpreted by general procedures. Through practice, knowledge compilation leads to automatization of the skills, particularly to task-specific automatic procedures.

Can Logo provide the "metacognitive experiences" necessary for such "thinking about thinking"? Flavell stated that such experiences are especially likely to occur in situations in which: people engage in and communicate about conscious cognition, particularly when they behave in new and unaccustomed ways; the outcome of what people think and do is important to them; people believe that their cognitions contain errors; and there are sufficient attention and memory resources (i.e., there is sufficient time to think about cognitions). Metacognitive experiences often are a serendipitous benefit of cognitive actions undertaken to achieve other cognitive goals. In Logo programming, children consciously solve self-selected problems using unfamiliar strategies. They must "communicate" their organization of the task and solution processes to each other, to the teacher, and to a machine. They must analyze their own thought processes for errors frequently and have adequate time to do so. Therefore, the occurrence of metacognitive experiences during Logo programming indeed appears likely. Characteristics of Logo programming that ameliorate problems in implementing programs to increase metacognitive skills will also be discussed.

Relevant Research and Educational Implications

Two major sections complete the presentation. First, several qualitative and quantitative studies on the utilization of componential abilities within the Logo environment, and transfer to other environments, will be reviewed. This will demonstrate that there is evidence that Logo may encourage the of utilization of both metacomponents and knowledge-acquisition components. Secondly, specific implications for constructing Logo environments that facilitate this type of development in elementary school children will be proffered.

Douglas H. Clements
Sheri Merriman
Kent State University

College of Education

401 White Hall

Kent, OH 44242

PROPOSAL FOR PRESENTATION AT LOGO '86

PRESENTERS: DR. ANNE PORTER JAWORSKI and CRAIG STIRTON

TITLE: INTRODUCING TEACHERS TO LOGO AS A TRANSFORMATIONAL ENVIRONMENT

Logo faces a serious dilemma. Critics suggest, with justification, that the initial claims for Logo are not being realized. Many proponents of Logo have developed excellent structured Logo curricula. Here students develop great competence in using the Logo language (Delclos & Littlefield 1984). However, in this structured environment, do they achieve the kinds of benefits as outlined by Papert in his philosophical treatment of the Logo environment? Other proponents have attempted to implement Logo through the discovery approach. Frequently this method has failed to demonstrate, using current measurement tools, either the depth of Logo knowledge (attributed to the structured model) or any significant transfer to other applications (Pea & Kurland, 1983). However, in both methods learners have consistently demonstrated similar levels of involvement in Logo activities (Kinzer, Littlefield, Delclos & Bransford, 1985).

We will draw upon a diverse body of research that indicates that engagement and the social environment are key factors in the learners development of skills that are transferable, that are measurable. Further, because of its particular characteristics that Logo is well suited to this orientation. This information has been incorporated into a theoretical in-service model that we call the "Transformational Environment." We will present the background and foundation of this model and suggest how it might be implemented so as to further develop the Perceive Potential of Logo.

Delclos, V. R. & Littlefield, J. "Does Logo Lead to Better Learning?" Presented at the spring conference of the Tennessee Association for Educational Data Systems, Nashville TN, April 1984.

Kinzer, C., Littlefield, J. Delclos, V. R. & Bransford J. D. "Different Logo Learning Environments and Mastery: Relationships Between Engagement and Learning," Computers in the Schools, Vol. 2 Nos. 2/3, Summer/Fall 1985.

Pea, R. D., & Kurland, D. M. "On the Cognitive Effects of Learning Computer Programming," Tech. Rep. No. 9, New York: Bank Street College of Education, Center for Children and Technology

Terrapin™
The Logo People
Terrapin, Inc., 222 Third St.
Cambridge, MA 02142 (617) 492-8816

A Panel On

TEACHERS' PERSPECTIVE OF LOGO USE IN THE SCHOOL
(Substitute r/evolution for use if you like)

This panel will give teachers some ideas regarding what has worked successfully with Logo in several schools, what hasn't worked and caused people to question Logo, and what has been done to address those concerns.

To start, each person will give a brief overview of how Logo is used in their school to put their comments in context. For instance, is it used in a class, a lab, or both; at what grade level is it used; for what purposes is it used.

Each person will cover the key successes of Logo in their school. They will also point out what factors they attribute this to, and whether they think the successes are replicable by others.

Each person will cover the problems Logo has had in their school, if it has had problems. If Logo was questioned, what was the focus of the questioning? How did Logo fare?

Each person will cover what can be done to address the problems raised in their school. Was it a problem of class/lab setup, of teacher training, of a need to fit it in the curriculum, or something else? Who needs to address these issues, publishers, administrators, or teachers?

The panel will consist of

Ricky Carter - Lesley College
Lynn Lieberman - Oakland Schools, Pontiac, Michigan
Beth Lowd - Lexington, MA Schools
Bettie Schwartz - Ladue Schools, St. Louis, Missouri

Jock McClees - Terrapin, Inc., moderator

Please address correspondence to:
Jock McClees
Terrapin, Inc.
222 Third St.
Cambridge, MA 02142
617-492-8816

Gary S. Stager
Paper Proposal: Logo '86

If there had been Logo I might have Learned Math

Whenever I attend a Logo conference or conduct Logo teacher training I hear countless revelations from Logo users about what lousy math students they were. These adults commonly admit that they previously disliked mathematics and that their use of Logo has created a new sense of interest and purpose for learning mathematics. I must confess that I was a very poor math student. Despite my programming ability, I steered away from majoring in computer science because of my distasteful and ineffective mathematics education.

Seymour Papert would suggest that many of us were "mathematically alienated". We were alienated because mathematics is abstract and we were expected to learn it through drill, instead of being immersed in a culture where we could learn mathematics in a natural, more concrete, manner. Some of us did learn mathematics, but most of us have forgotten what we learned. Mathematics was presented as something that was hard, cold, and externalized from our own experiences. As students, we saw little meaning or practical applications for learning math.

Logo provides the student with an educational microworld in which they can explore and manipulate many abstract mathematical concepts. The child can "mess about" with this newly discovered knowledge and in a short time realize that these simple, powerful ideas can be combined to produce complexity. With Logo, the child becomes deeply involved in his/her learning process and develops a learning style. The student's curiosity and new-found interest will enable him/her to discover new concepts by the manipulating what they already know .

Everybody knows that turtle graphics has many educational benefits for the child. By teaching the turtle to draw a figure, the student must assume the role of the turtle in his/her own space and reflect on how they themselelves would draw the figure. This process encourages problem-solving and reinforces mathematical concepts inherent in the particular programming activity. The child takes control of their own learning and establishes a personal relationship with mathematical ideas.

Turtle geometry provides a rich "mathland" in which movement, measurement, turning, arithmetic operations, logic, and many other mathematically powerful ideas can be explored. Students programming with turtle graphics are bound to gain an understanding of numerous geometric issues.

The Presentation:

Although, turtle geometry has infinite educational possibilities, there are other areas of the mathematics curriculum that can be addressed by Logo. In this presentation, I intend to show several Logo math microworlds that I have developed. These Logo procedures attempt to address issues dealt with in the (K-8) elementary school mathematics curriculum, including:

Fractions: Microcomputers do not generally deal with fractions, although this is a consistent source of difficulty for many students. I will demonstrate math tools that allow you to manipulate numerators and denominators, add, subtract, multiply, and divide fractions, and reduce fractions to their lowest terms. There are also procedures for simplifying mixed numerals and fraction-decimal conversions.

New Math Operations: When you attempt to solve mathematical problems, in Logo, you realize thot there are mathematical operations that need to be added to Logo. Some of these operations exist in other versions of Logo, but I wrote them for Apple Logo I. Many of the following new math operations are of particular use to students in the elementary school: Absolute value (ABS), the distance between any two points (Distance), the ability to use exponents (POWER), division of real numbers (DIVIDE), (DIFFERENCE) between two numbers, and a procedure that lists all of the factors of a number (FACTORS).

New Math Predicates: Logo predicates test a condition and return a value of true or false. They are of paramount importance and many new ones had to be created, including: POSITIVEP, NEGATIVEP, ODDP, EVENP, PRIMEP, COMPOSITEP, INTP (is the value an integer?), and FACTORP (is one value a factor of another value?).

Baby Turtles: This is a set of turtle graphic microworlds designed for the very young or handicapped child.

The Counting Machine: This is a procedure that allows the child to ask the computer to count up or down, between any two numbers, by any increment or decrement. This is useful for learning addition, subtraction, and counting.

The Perimeter Tracer: The perimeter tracer is smarter than the average turtle. As this turtle draws, it keeps track of how many steps it has taken. Very useful for learning distance, perimeter, and area.

The Pie Chart Procedures: These procedures will take a list of numbers, either inputted by the user or by another procedure, and draws a pie chart with each "slice" in a different color. The percentage of each "slice" to the whole will be noted.

The Symmetry Toys: These are procedures that allow the student to simulate multiple turtles by drawing reflections of the turtle's movement in two-four other quadrants. This provides a microworld for "messing about" with symmetry.

During the course of this presentation I may share with you many other Logo math tools that can be used by the child to explore: the cartesian coordiante system, positive and negative numbers, decimals, probability and statistics, similarity, pi, circumference, radius, diameter, the total turtle trip theorem, and many other concepts. I have created these math tools for and with my students, and I will present examples of their work.

I have also explored Seymour Papert's ideas of "thinking mathematically" and "mathematics as a language". As a result of these ideas, I have created a microworld in which primary grade students can express mathematical ideas in their own words. A first grader might say, "How much is twice 3 more than 5 takeaway 2?" The computer would then return an answer and might even ask the child to solve a similar problem.

I will also demonstrate how word problems may be solved in a similar way by older students. The notion of simple procedural machines that take a value as input, operate on that value, and output a new value is a very powerful idea. A student could create a series of one and two line procedures that could be used to solve several different problems, for example: "How many 3.5 inch thick books will fit on a 7 foot shelf?" As you can see, the problem is expressed in "mathematical language" and must convert 7 feet to inches and divide 3.5 inches into 7 feet.

Conclusion:

The power of all of these Logo math tools and microworlds is that they are all flexible and may be used in conjunction with any other types of computation. This truly provides the student with a microworld in which simple powerful ideas are cabable of being developed into much more complex concepts.

It is my hope that these tools will create interest in mathematics and stimulate the child's imagination and curiosity. Maybe the next generation of boys and girls will be turned on to mathematics and will perceive it as meaningful, challenging, and powerful. Mathematics is something that can be learned and enjoyed by children, possibly avoiding another generation of "mathematically alienated adults". I believe that Logo can signifigantly help us towards this goal.

Respectfully Submitted:
Gary S. Stager
12 Locust Place
Wayne, N.J. 07470

Ricky Carter
Lesley College

The Logo Algebra Curriculum Project
(proposal for presentation at Logo '86)

Logo's entry into elemntary schools has been mostly under the guise of computer literacy or the felt need that all students should learn to program a computer. As of late the educational world seemsto be turning away from computer literacy as an isolated curriculum topic, and as "tools" become the rage many elementary schools also seem to be turning away from teaching programming. Given these trends where will Logo find a place in schools? Perhaps it can be used as a force to directly reshape the way particular school topics are taught. In much the way turtle geometry is an attempt to provide a different model for learning geometry, perhaps other topics could be rethought and restructured through a Logo based approach. Last year we recieved an NSF grant to develop an algebra curriculum for sixth grade students using Logo. Our hope is to prove that we can effectively teach sixth grade students many of the ideas of first year algebra through a curriculum that involves Logo programming and a series of Logo based manipulative environments.

This year we have been developing materials and doing some prepiloting in a number of local schools. Currently we have developed three curriculum modules:

1. From English to Algebra
In which students move from writing Logo programs that generate random gossip to writing programs that will generate algebra quizzes.

2. Marble Bag Stories
In which students learn to translate number puzzles into algebra notation and to solve equations using an iconic representation system of marbles and marble bags.

3. Codes and Functions
In which students learn to write Logo programs to code and decode secret messages as an introduction to functions, and to write Logo programs to create secret numerical function machines for other students to try to solve.

The Logo Algebra Project

Of the many problems encountered by students new to algebra the abstractness of the new notation and the need to combine both new strategic and new arithmetic processes when solving problems seem preeminent to us. One strategy is to involve students in learning about algebra by having them write programs that generate algebraic problems We are also developing two iconic labs that will allow students to represent and manipulate algebraic ideas using graphic icons (marble bags and function machines), and then help them to translate these representations into formal notation. A third lab is also planned that will function as a symbol manipulation workbench where students will be able to practice the strategic side of algebraic manipulation while the machine takes care of the arithmetic side. All of these will be written in Logo. Although this year many of the activities have been successfully tried by teachers using Apple II's, the icon lab environments have needed the graphic capabilities of the Apple Macintosh. We are developing our icon lab software on the Mac and we will be using Logo on Macintoshes in our pilot study next year.

We would like to share our ideas, and our experiences at Logo '86. We would plan to present our work to date, to discuss our attempts to simplifying the amount of Logo syntax and grammar students have had to master in order to use Logo programming as a tool for learning Algebra, and to share some of the difficulties (and successes) we have had in trying to create activities that are personally engaging and involve student initiated ideas.

PRESENTATION ... LOGO 86

A PAPER ON THE...LEARNING MATH WITH LOGO...PROJECT

...RUDY V. NEUFELD ...LONDON BOARD OF EDUCATION

SUMMARY OF THE OBJECTIVES EMPHASIZED IN THE PROJECT

LOGO is used as a gentle introduction to computer literacy as well as a help in implementing some of the Gr 4-7 mathematics curriculum. The project material oncourages students to use a guided discovery approach in order to investigate, learn and understand mathematics. With LOGO in the mathematics classroom, an abstract concept takes on concrete meaning.

A number takes on meaning when it results in some movement on the screen. LOGO clearly shows the relationship between mathematics and spatial ability. This not only helps the regular student but also the special and the remedial student.

PROBLEM SOLVING is emphasized in this project. In LOGOLAND the student becomes the teacher of an obedient student, a robot called a turtle. The student in this role, is in control and is encouraged to take risks. Every plan does not always work smoothly but it may give rise to other ideas. This learning environment meets the need for true, open-ended discovery. With LOGO the user learns about learning itself.

The STUDENT MATERIAL can be used as part of a class lesson, or can be used individually by the student. The format is such that few computers are required. In most cases a GENTLE INTRODUCTION and a PRE-COMPUTER LESSON preceeds a GUIDED DISCOVERY ACTIVITY which is then followed by an EXTENDED LEARNING ACTIVITY. In the guided discovery exercises, students are encouraged to ...THINK THROUGH, WALK THROUGH, WRITE DOWN COMMANDS ... BEFORE using the computer and commanding the turtle on the screen. This is following by TALKING ABOUT WHAT and WHY something happened and some reinforcement exercises to solidify the concept.

The TEACHER RESOURCE MATERIAL gives clear chapter and section objectives. At the beginning of each section, a brief summary of LOGO commands and skills required and learned is given. A list of mathematical concepts learned and materials required in the section is also given. The depth of coverage of material is clearly delineated for grade 4,5,6,7. Suggestions for a
PRE-COMPUTER LESSON, GUIDED DISCOVERY ACTIVITY and EXTENDED LEARNING ACTIVITY are outlined. Suggested answers to many exercise questions in the student material are given. A collection of BLACKLINE MASTERS to be used with activities in the student text are given.

Examples and exercises are based on the grade 4,5,6,7 mathematics curriculum. Students do not need to be able to do all the work in a section before going on the next section. As students learn more mathematics and/or more LOGO, they will be able to do more exercises and/or do old exercises in another way.

THE 4 STEPS IN AN INTRODUCTION TO LOGO AS A TOOL IN CURRICULUM:

In the project, 4 gentle and logical steps in an introduction to LOGO and to its role within present curriculum are used:

STEP 1...THE FLOOR ROBOT gives a real feel for geometric shapes and direction. This is an important step for learners at all levels. It prepares the learner for Step 2. When the learner has difficulties with the shapes on the screen, it is natural to go back to walking through the design on the floor or on paper.

STEP 2...THE JUMP TO THE SCREEN introduces a code that is simple enough for non-readers to use but is powerful enough to discover and investigate mathematical ideas. This step is a smooth transition between the turtle on the floor and the turtle in LOGOLAND.

STEP 3...STEPS INTO LOGOLAND investigates some LOGO commands. It also makes sure that the user is familiar with the turns and calculations in LOGOLAND. Angle patterns in various shapes are investigated. Step 3 creates a solid base for following concepts...Step 4.

STEP 4...THE TURTLE IN SCHOOL will be taught lessons (PROCEDURES) by the user ... the teacher of the turtle ... the person in control. This teacher must thoroughly understand a lesson and present it in a logical way to the student ... the turtle. The teacher will be open to student (turtle) reactions and then adapt the lesson. The teacher will learn math concepts and problem solving methods by teaching and then testing the turtle!

The role of the computer is de-emphasised. The emphasis is on the user being in control of an obedient turtle. The user is encouraged to investigate and take risks ... the turtle will react to teacher lessons and help the teacher learn ... A POSITIVE LEARNING PARTNERSHIP!

THE CLASSROOM TEACHER:

1. Facilitates in a shared learning experience.
2. Is open to letting the unexpected happen.
3. Encourages students to:
   ...release their curiosity.
   ...view a "bug" as an opportunity to learn.
   ...break a complex problem into manageable components.
   ...play turtle.
   ...search for patterns and relationships.
4. Is not limited by a few computers.
5. Uses other concrete materials as well as LOGO.
6. Has fun and learns with the students!

MY PRESENTATION:

A) Use 4 informative and colorful posters to show the 4 steps through LOGOLAND.

B) Briefly demonstrate some Turtle-Trainer helpers:
SLOGO, ROBOT TURNER, PENCIL ROBOT
(much LOGO can be done off the computer)

C) A Turtletrip Through Squares

A sequence of mini lessons on using LOGO to help in understanding the square. These lessons will summarize and clarify some of the objectives listed in the preamble. I will begin with the square in primary classes to the square in high school classes. In each of these lessons the problem solving capabilities of LOGO will be emphasized.

...The repeating pattern
...Square and Square Plus
...The turtle learns how to square
...Squares grow ... variables
...Squarral
...Gymnastics ... squares flip, turn and slide
...Fill squares...Fill triangles...trapezoids
...The square wheel
...

Thank You

Rudy V. Neufeld
7 Conifer Cres.,
London, Ontario
N6K 2V3
Canada

TITLE LOGO AS A TOOL TO DEVELOP THINKING SKILLS: WORKING WITH TEACHERS

PRESENTER Marilyn Schaffer, Director of Educational Computing
University of Hartford, West Hartford, CT 06117
(203) 243-4277

PRESENTATION FORMAT

Using videotapes and a computer, the presenter will describe and demonstrate a teacher training sequence that was designed to assist educators to explore the uses of Logo and related materials to enhance students' creativity and higher level thinking abilities.

PRESENTATION CONTENT

The presentation will focus upon a teacher training project that is being developed to enable educators to explore the uses of Logo microworlds in the classroom to serve as vehicles to facilitate student problem solving. The major theme of the training is that Logo, in contrast to traditional classroom activities, can present students with the following types of problems to solve: "what will the computer do under certain conditions?"; "why or how did the computer do a particular thing?"; and "what should I 'tell' the computer so that it will do what I want it to?"

The presenter will share with the audience a sample of the training strategies developed during the program, based upon this theme, as well as the activities used by the teachers in their classrooms. Some of the work accomplished by the students will be demonstrated on computer and videotape.

The training activities are based upon the premise that, because of the affective and cognitive flexibility of young students, as well as their potential creativity and curiosity, the early school years are a profitable time to expose children to the problem solving environment inherent in Logo.

The construct of "problem solving" will be defined and discussed in terms of student behaviors observed by the teachers as the children are involved in Logo interactions, including the following examples: predicting an event; hypothesizing why a particular outcome occurred ; changing one variable while keeping other conditions constant; focusing attention on one factor at a time; and drawing analogies from one experience to another.

INTENDED AUDIENCE

This presentation is directed toward an audience of teacher educators and teachers, working with children ranging from preschool through elementary school. The presentation is not limited to any audience size.

A New Approach for Logo Teacher Training

by
Ted J. Brucker and Carol Scheuer
George Washington High School
655 South Monaco Parkway
Denver, Colorado 80224
(303)-399-7770

One of the problems facing the Logo community is how to get competent instruction in Logo to the student in the elementary classroom. The traditional approach of providing classes or inservice instruction for elementary school teachers, who then teach Logo to their students, has proven to have a number of pitfalls.

First, there frequently is a lack of available personnel. Many teachers are overburdened with other teaching chores. Some are fearful and feel they lack the proper "background." Some, initially excited and interested, cool to the ideal after talking to colleagues who have´been through less-than-effective training courses.

More problems await those who do take traditional training in Logo. Because of a chronic lack of time and personnel the instructor may have only a little more experience and training in computer usage and Logo than those taking the class. Sometimes the instructor is well versed in programming and computing, but knows almost nothing about Logo, its underlying philosophy of learning, or its power. The time allotted for such training is usually too limited, especially if computer usage, including DOS and programming concepts, must be taught. If the training is given after the school day, teacher fatigue compounds the above problems.

As a result, many students in the classroom do not receive competent computer instruction. At best their instruction is inadequate and incomplete. At worst, the experience is one of frustration and misinformation that totally sours students on the whole computer experience. From a Logo point of view, such instruction totally misrepresents Logo, its purpose, and its ability to lead students into new worlds of exploration and learning.

At George Washington High School in Denver we have found a new way to get competent Logo instruction into the elementary schools. Instead of offering training to classroom teachers, we offer to the elementary schools, their teachers and students, the services of well-trained high school student-instructors. This provides a number of advantages.

First, the student-instructors are well trained in computer usage and programming. Most already have had courses in Pascal, FORTRAN, BASIC, C or Forth. Many also have had experience with integrated software application packages, word processors and electronic spreadsheets. All student-instructors are given advanced training in Logo, including instruction on tail and non-tail recursion and on the philosophy of Logo. Secondly, these student-instructors are highly motivated and enthusiastic, and this attitude is infectious. Finally, these student-instructors are excellent mentors for the elementary students.

For the past two years we have tried two types of delivery systems. The first has student-instructors going into the elementary school and teaching the students in their regular classroom. The elementary students learn in familiar surroundings and on equipment they have available during the regular school day. Some elementary teachers who have been opposed to or disinterested in getting computer training become interested when they see their own students so excited and animated. Frequently our student-instructors have been asked to help build a computer program for the entire school, including teacher training, after elementary teachers and administrators see the effects of a well taught Logo class.

The second way of providing Logo instruction to these students involves the elementary school students being brought to our high school by interested parents for instruction before the school day begins. This method is usually used when the elementary school lacks the necessary computer equipment or the school does not yet feel comfortable with high school student instructors. Using this system the elementary students have the advantage of working in a modern well-equipped computer lab, in a high school learning environment. These students then return to their elementary schools and become "experts." As such they become an important resource for other students and teachers.

Both methods have proven highly useful and effective. The most serious problem to date has been the difficulty in getting a sufficient number of high school students available for such work. At the moment we have more success that we can adequately handle.

We do not suggest that this is the only way to get good Logo instruction into the elementary classroom. But, this type of grassroots approach has proven to be a successful alternative to more traditional methods for providing effective computer instruction. These high-school student instructors embody the enthusiasm and desire for exploration and learning that Logo first promised.

[Ted Brucker is the faculty organizer and instructor for the program. Carol Scheuer is a 12th grader who has taught the in-building portion of the program this past year. Our presentation includes a videotaped presentation. Samples of curriculum and student work will be provided.]

Logo... the writing should be on the wall!

Lynda Colgan

Wall. A simple four letter word. One that is the focus of a rich collection of figurative language! Metaphors and similes abound! How often have we heard the expression, "*the writing is on the wall*"? How often have we been told that our latest and greatest idea could only be described as being "*off the wall*" ? People who are forced to conform to rigid sets of rules often feel "*walled in*". A person reluctant to respond to change is said to "*build walls*" around himself. The rock band Pink Floyd created an entire musical commentary on contemporary education and society metaphorically titled The Wall The list could go on and on.

And what do walls have to do with LOGO? **Everything**. Computer-using leaders recognize that LOGO has the potential for realizing the dreams of which true education are made. Educators have been touting activity-based, discovery-oriented, accessible, wholistic learning for years. In the philosophy and structure of LOGO they recognize that this model is made accessible! To them, the writing is on the wall. More importantly, they recognize that LOGO is the very brick and mortar with which the process of constructing the great wall of knowledge can begin - an awesome, winding, creative, diverse structure with no end in sight. However, there are equally as many educators who have used the same raw materials to build a very different wall. They have manoeuvred LOGO into building prison walls around itself. The prison walls of curricularization, scope-and-sequencing and compartmentalization. And lastly, there is the myriad of educators who have erected great fortresses around themselves and their classrooms in an attempt to keep LOGO out or at least reduce its status to that of "just another brick in the wall." They shun the underlying theory and philosophy. *"Imagine those ivory-tower types and their idea that through programming computers properly, everything else can be enhanced!"* To them, LOGO is impractical, improbable: off the wall!

The walls are much more than symbolic. To teacher trainers, they are frighteningly tangible and equally as formidable. How can we respond to the challenge?

A tongue-in-cheek illustration suggests one approach:

Perhaps the special breed of quality LOGO-using educators should become organized into a subversive movement. Imagine a mass meeting, thousands of LOGOPHILES gathered in one room to receive their mission statement: Go forth and cover the walls of educational institutions everywhere with graffitti! Classroom anecdotes. Meaningful quotations. "Neat" super-procedure project ideas. An eclectic LOGO-coloured rainbow. After all, it has been said that the medium is the message, and studies suggest that graffitti can be both a powerful forum and a moving art form. It is time that the "LOGO Panthers" **get the writing on the wall**. Perhaps then the powerful message will at long last be read by the non-LOGO using educator, who will become curious enough to take his first turtlesteps!

Having accomplished this first mission, perhaps the LOGOPHILES will accept another challenge. A true Mission: Impossible. This time, walls are to be demolished. Not just any simple walls, but the monumental fortresses that teachers and teachers-in-training build in response to the perceived need to control LOGO or keep it out altogether! The ammunition? LOGO truths hurled at the bastions!

The first wall to be under siege will be that constructed in an attempt to contain LOGO within a box comprised of six simple walls. Feeling that they could not simply record an entry of **thinking or what if???** into a daybook of lesson plans, many teachers attempted to pre-package LOGO into a series of curricularized units:

- Today's math lesson will be on regular polygons. We will use LOGO to draw them."
- "Tomorrow's art lesson will challenge students to create imaginary animals using LOGO REPEAT procedures."
- "Our work with LOGO is finished for this year because we must not go beyond our current grade-level curriculum."

Teachers reacted instead of acted. After all, when microcomputers and simple C.A.I. software were introduced to schools, teachers everywhere praised.the advent of so powerful a teaching tool! Students' work could be individualized! There would not have to be uniformity in math or spelling or any other core curriculum area, for that matter. *(Yet did that happen? Quite simply, no. Most C.A.I. software is grade- and domain-specific. While students could most certainly pursue slightly varying paths along the subtraction trail, for example, the teacher could always monitor the route each travelled from a master observatory. And besides, keeping track of these students was not too challenging because there were a limited number of paths and predictable content along each.)* There was a feeling of panic when it was acknowledged that LOGO did not fit the pattern. Here was a computer application that was not only grade-free, ability- and subject-free, but based upon a philosophy of "fixability" and transparency. It encouraged questions that did not fit the mould: Jacqueline wants to know if LOGO has IF primitives! Scott wants to know if he can use trigonometry to solve his problem! Sarah is still using a clock to help her determine rotation angles. This amount of disparity simply could not be accommodated in one classroom, unless nuggets of LOGO were selected for specific curriculum applications. And so up went the walls - boxing LOGO in!

The walls constructed to preserve the role of teacher as "director of the learning process" will be next under attack. LOGO undeniably defies the traditional teacher/student relationship and proposes that the classroom be transformed from a centre of **teaching** to a centre of **learning**. The first step in the metamorphosis will be that the teacher will be challenged to assume a new role: not that of keeper of the knowledge, but that of learning facilitator, and more importantly, co-learner The Queen's University Faculty of Education study on the creative use of computers in education reports that this is one of the most difficult transitions for teachers to make. The study reports that it is difficult for many educators to accept joint learning experiences. Many are comfortable in the conventional mode of "being in charge" and have difficulty in dealing with the phenomenon *of being surpassed by students in their knowledge and ability in computer-based learning.* They are strongly tempted to continue to direct learning via assigned tasks. The response? The walls that are now so difficult to demolish.

Having accepted that graffitti and a militant green-bereted LOGO version of the Guardian Angels are probably not the answer, how do we proceed?

Teacher trainers must take a firm approach to LOGO education. First of all, there must be no apologies for the fact that LOGO is indeed a programming language. Secondly, there must be no attempt to simplify LOGO by sidestepping its philosophical roots and related pedagogical imperatives. Superseding both objectives must be a commitment to affording teachers the opportunity to explore and discover, and to be actively involved in their own learning.

An interesting first session would be to insist that each teacher/candidate attend with a "real live" student: an elementary or secondary school learning partner. The first segment of the session would involve an in-depth look at a range of C.A.I. software such as <u>The Milliken Math Series</u>, <u>Dictionary Dog</u>, or <u>Dinosaur Dig</u> followed by an analysis of the software's impact on learning. The microcomputer is touted to be fast, dynamic, interactive, motivational, and a link to the devlopment of higher order thinking skills. Does this software concretize these characteristics? Having looked at software from the instructional paradigm, the next step would be to have the "odd couple" candidates boot LOGO. They will soon discover that this software is different. *"It doesn't look like a turtle!" "Is it just going to sit there, or what?" "It says it doesn't know how to "*!Bo□¶!!!!"* Within a few moments, given a preliminary tool kit of basic primitives, and "problems" to be solved, these pairs will undoubtedly discover (though probably will not articulate) the basic aims of microworld investigations - accessible, simple, syntonic learning tasks. With LOGO the microcomputer escapes from the instructional paradigm and enters that of emancipation - freedom to explore and create and engage in computer-assisted serendipity! The significance here, is that no one "spelled out" the face-value benefits of LOGO. The learners were left to make their own conclusions, and with ownership of an idea, comes approval *(after all, don't we all like our own ideas?!)* and a joy in seeing it come to life.

Subsequent sessions must be designed with challenges to teachers and students that are tangible evidence of learning through LOGO. For example, Write a LOGO program that will draw a circle within a square of any given size such that the circle is tangent to the sides of the square. A simple-sounding problem, and yet, the originator of the problem, Robs Muir (NLX) reports that in the course of their explorations, many problem-solvers unwittingly derived the value of **pi**. As Muir states, "You learn something old every day!" Another project could be the creation of an interactive "Game Show" (Feurzeig) in which students design questions and construct their own solution algorithms in order to create a quiz that will test their peers' knowledge of anything from sports to a particular algebraic concept. The significance again, is that there is not an attempt to curricularize LOGO, but to develop a series of nuggets upon which sound LOGO-based explorations lead to the discovery of a concept. More important than the actual challenges, will be the emphasis in these sessions on the strategies the students used. They will be asked to articulate their method of attacking the problem, outline the resources they required and how they were obtained, and react to the importance of teamwork and co-operation. In this way, teachers will be experiencing the strategies that they themselves will transfer to the implementation of a LOGO program of their own. The LOGO training? Explicit. The pedagogy and structure? Implicit.

By living the LOGO experience, teachers and students will draw their own conclusions. They will question what LOGO can do to them and for them. Theodore Sizer analyzed many American schools and concluded that not enough of our students are hungry to learn. The same is undoubtedly true of our teachers. These co-operative sessions have two objectives: to whet the appetites of students and teachers alike by introducing them to exciting learning environments, and to alleviate the empty (hungry?) symptoms of the stigma diagnosed as being "teacher as learner." When there can he first-hand experience of the joy shared by teachers and students in a co-operative learning environment, there will be a perceived need to introduce LOGO into the regular spectrum of classroom activities in order to improve rapport and enhance the classroom microworld.

The next step could be to introduce informal dialogue sessions with LOGO-using teachers and students and more importantly, **participatory** visits to exciting action-packed classrooms. It must be emphasized that these are not LOGO expert to LOGO novice sessions, but casual, "Let me tell you about the day!" or ""Can you believe my students did this?" sessions. The purpose? To teach the significant part of the philosophy-in-action by role-models and examples. After all, seeing is believing. Perhaps they could talk to teachers who long to recapture that first hour they used LOGO in the classroom, excitedly rushing to the boot the disk without reading the manual, and recalling the feverish conversation: "Now, how do we move this thing?" "Wow! Look at that!" "Oops! Where did the turtle go?" "What shall we do next?" Everyone laughed, but everyone learned - together! Perhaps they could exchange ideas with a few dedicated LOGO-using educators who have been introducing their students to the wonderful world of **what if???** afforded by LOGO investigations. These rare individuals live the philosophy. They use the computer fueled with LOGO not only as a tool, but as an environment. And in so doing prove Neil Postman's hypothesis *that children enter schools as question marks and leave as periods* false! Perhaps they could talk to the fortunate students in unique learning environments who share in these rich experiences. They inquire. ... .explore. ... .discover. ... .debug. ... converse. ... ...co-operate. ....and make transfers. They are knowledgeable about the structure of the programming language and use it comfortably, competently and confidently. But they look beyond square brackets and quotation marks. Why? **Their teacher has never concentrated upon keystrokes. The emphasis has been upon mindstrokes!** These students know that the gift of **"what if"** is theirs!

Via the witness and exchange of ideas, and active participation, the teacher-learners will be encouraged to discover that these LOGO-using teachers and students dare to translate educational theory into practice. The LOGO educator understands that in principle and in practice there is no need to organize students into "packages". Why should a third grader not study trapezoids and parallelograms if he makes an interesting discovery just because they are Grade five curriculum material? Why must textbooks always be completed sequentially? Why should apparantly unrelated curriculum areas not be married? Do such unions not often result in the creation of highly unique hybrid offspring? These same educators understand that "computers in education" does not translate into a hardware/software interaction, but that the computer, LOGO and student form a complex networked triad that supersedes symbols agraphics on the screen. As the experienced LOGO-using educators talk about and live their personal philosophy of education, perhaps the new teacher-learners will be encouraged to re-examine their own pedagogy and how it directly translates to their classroom and students. And that self-examination and introspection, coupled with the ideas and strategies that they have acquired, will make them ripe for other lessons about other educational philosophers and how their pedagogical notions can be translated into action. If we can encourage teachers to go beyond the keystrokes in their training, and concentrate upon mindstrokes such as heuristic learning and remodelling of classrooms to reflect the Einsteinian paradigm, the students are most certain to benefit...learning will occur outside subject walls via discussion, teamwork and co-operative endeavour.

A second phase in the implementation of this project would be to design a new dimension to the training. For this segment of the project, the teacher and student would attend with a new learning partner - the Principal, Vice-Principal or Supervisory Officer. The enthusiastic newcomers to LOGO could then become the "Computer Coaches" giving assistance to the administrator as he learns the language and concomitantly guiding him/her through the same range of discovery activities that illustrate the power of the use of open software. Without the support of administrators, implementation of a LOGO program will be difficult, if not impossible; and without having shared the experience, those in positions of responsibility will not be fully aware of the requirements, potentials and complexities of such a program.

Of course, there are other components to teacher training. There must be on-going support systems built within schools and within boards of education to accommodate trouble-shooting for beginners and extensions for the more experienced. "Life-long learning" must become the slogan! Teachers who are just beginning to explore the possibilities of LOGO must be encouraged to read journals and periodicals. This means that editors must assume responsibility for the content of their publications and strive to include articles that describe paradigms outlining new models for learning and present LOGO challenges that encourage analysis, interpretation and communication. As well articles must explore the full potential of LOGO-based units and courses of study and suggest strategies for achieving their success. As curriculum guidelines are revised and updated, Problem Solving and Thinking must be included, perhaps including a LOGO element. Provisions must be made for teachers to attend conferences and workshops. Organizers of such meetings must attempt to meet the needs of the audience via sessions of varying levels of difficulty, and more importantly choose speakers who are dynamic and whose style reflects the philosophy that must be conveyed genuinely and effectively. Electronic bulletin board systems should be used to their full potential as a forum for questions and discussions. While all of these supplementary activities will undoubtedly be very important and have great benefit, the greatest teacher training of all will be in the "WOW's!!!" heard in classrooms as teachers and students **do** LOGO and share their unique approaches and ideas.

In an article entitled Learning in the Global Village, Marshall McLuhan and George Leonard begin with the following statement: The time is coming, if not already here, when children can learn far more, far faster in the outside world than within schoolhouse walls. Perhaps with appropriate teacher training and implementation, LOGO will help teachers to make the activities **inside** schoolhouse walls more vital. After all, when teachers learn about LOGO they not only learn about computers, but about education.

```
7TO BREAK.DOWN.WALLS
>MAKE "TEACHER.TRAINING "FOSTER PROCESS
>MAKE "CLASSROOM "STUDENT.CENTRED
>REPEAT FOREVER[CO-OPERATIVE LEARNING]
>MAKE "TEACHER.SUPPORT "ONGOING.ACCESSIBLE
>BREAK.DOWN.WALLS
>END
```

Building Mathematical Systems

Using OUTPUT to build functions from
elementary word problems through high-school math

Mathematics, even at the elementary school level, positively depends on the concept of functions: rules that map some set of inputs onto a set of outputs. Yet the elegant relationship between Logo "processors" (operations with inputs) and mathematical functions tends to be underexplored because OUTPUT is seen as too hard for students to learn. My paper and talk about "Iconic Programming" describes a language and imagery that has been quite effective in teaching students how to write and use processors. The purpose of the paper you are now reading is to give examples of the development and use of systems of functions in a few mathematical domains.

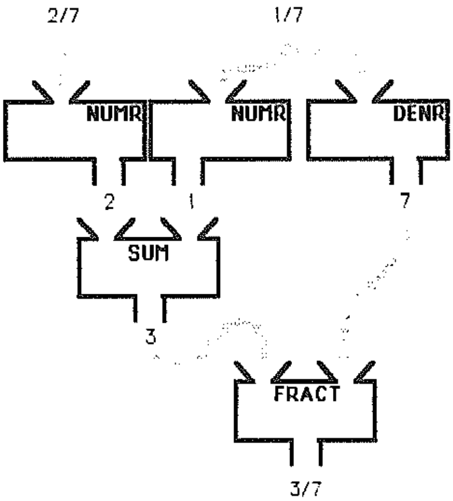
"Working models" of a mathematical system allow students to conduct mathematical experiments. By building these models themselves, students get to explore the interrelationships of the parts of a system and get to participate in the uniquely mathematical experience of making up the rules and then seeing how they play out.

Let's first consider a high-school project to develop a fraction calculator. At the surface, the student is building a system of procedures to manipulate rational numbers exactly rather than relying, as computers generally do, on decimal approximations. Below the surface, the student who builds such a system—procedures to add, subtract, multiply, and divide fractions, and to reduce fractions to lowest terms—is investigating the structure of algorithms, and is engaged in a study of the properties of mathematical systems.

But why choose fractions and not some new mathematical territory for a high-school project? With the same techniques, students could be writing a calculator for complex numbers, vectors, or some other "new" mathematical territory. Indeed, these may be perfect follow-up projects, and may even be the intended aim in the course, but in their first exposure to the business of creating, rather than merely learning and using, a mathematical system, letting them create one they already know well focusses their attention on algorithm, per se, rather than on the particulars of a new and only partially familiar algorithm. Students who are first learning fractions could also build the fraction calculator, but ideally it would not be their first project building working systems of functions.

Here is one approach to building one of the functions in the fraction calculator, a procedure that adds two fractions that have like denominators and expresses the result in unreduced form.

We can invent a process for adding two fractions that have like denominators. We will need to add the two numerators to produce the new numerator. The new denominator is the same as the old. So, we will need a machine that can extract the numerator of a fraction (e.g., getting the 2 from 2/7), and a machine that can extract the denominator (e.g., getting the 7). We will also need a machine that can add integers (e.g., 2 + 1), and a machine can combine integers into a fraction, (e.g., combining the new numerator 3, and the new denominator 7 into the fraction 2/7).



We have many options for representing the fraction, but the simplest probably is a list containing the numerator and the denominator (e.g., [2 7]). If we choose that notation, then it is quite easy to design the necessary machines:

```
TO NUMR :FR              TO FRACT :NUM :DEN
 OP ITEM 1 :FR             OP LIST :NUM :DEN
END                      END
```

and the Logo expression for the computation (the assembled set of machines) looks like this:

```
FRACT SUM NUMR [2 7] NUMR [1 7] DENR [1 7]
```

The iconic programming suggests some tool procedures—FRACT, NUMR, and DENR—that we will need, and tells us how to assemble them, but we have yet to build the function we originally intended to build.

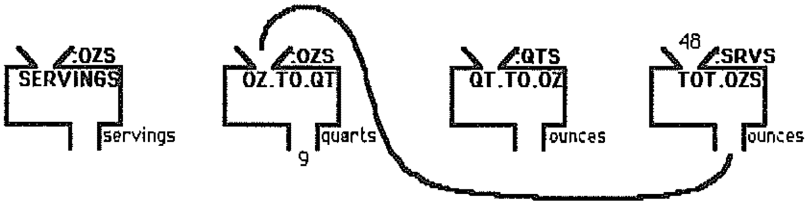
To package this method into a single machine that is capable of adding two fractions with like denominators, we need to design the look of the machine. It needs two input hoppers, one for each fraction, and it needs an output spout for the sum. If we call it ADDLIKE and call its hoppers :F1 and :F2, the rest of its definition follows naturally from the example:

:F1
:F2
ADDLIKE

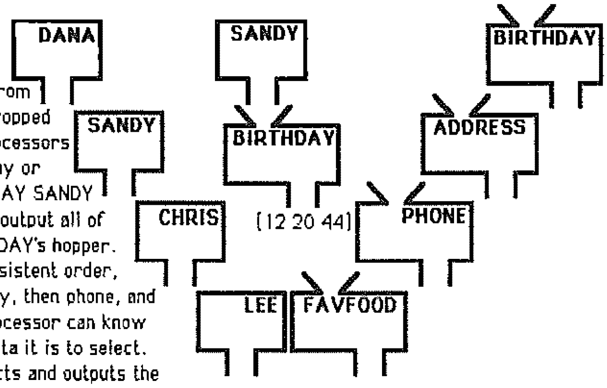
```
TO ADDLIKE :F1 :F2
 OUTPUT FRACT SUM NUMR :F1 NUMR :F2  DENR :F2
END
```

This model is versatile: The programming structures needed for multiplying 3x3 matrices are no more complex than the ones used in the simplest version of the fraction calculator. And the fraction project is easily extended. An extension that I particularly like because it introduces a kind of recursion that my students never seem to find magical is this: Fix your fraction procedures so that they can handle fractions that have fractions, instead of integers, as numerators and denominators. A half a fifth ($^{1/2}/5$) plus a half a fifth ought to be a whole fifth (1/5), and so forth: ADDLIKE [[1 2] 5] [[1 2] 5].

For younger students, mathematical modelling may involve the creation of small systems that extend simple word problems. For example, Dana will need 48 servings of juice for a party. If each serving is 6 ounces, how many quarts will Dana need? As it is posed, the problem has one answer. But let us suppose that Dana were generally responsible for class parties. It might be useful to have a system that could recalculate for different sized drinks, different numbers of people, different expected servings per person, etc. Students may be encouraged to think of little calculating machines that, "if only they had them," would do all the work. The figure below shows a small set of machines, brainstormed for the unelaborated version of this word problem. A line shows the path that answers Dana's original need.



What else can we design? An excursion outside of mathematics suggests some new techniques that are valuable back in conventional math.



What else can we design? A data base with mix and match sources and processors. Data from any one of the sources can be dropped into the hopper of any of the processors to find out, e.g., Sandy's birthday or Lee's favorite food. PR BIRTHDAY SANDY causes the procedure SANDY to output all of its data about Sandy into BIRTHDAY's hopper. If source data is stored in a consistent order, e.g., address first, then birthday, then phone, and favorite food last, then each processor can know exactly where to look for the data it is to select. In this example, BIRTHDAY selects and outputs the second item in the list that is dumped in its hopper. Two of the procedures might look like this:

```
TO BIRTHDAY :WHO      TO SANDY
 OP ITEM 2 :WHO         OP [ [[Williams St][Elizabeth NJ]] [12 20 44] [none] [pizza]]
END                   END
```

If we have organized well enough, we can create processors that further refine the selectivity of the data base by extracting, e.g., the birthyear: OUTPUT ITEM 3 BIRTHDATE :WHO

This project, I might add, has considerable value in its own right, beyond its utility as a bridge to a new mathematical territory. The organization of information, as in the list in the SANDY procedure, is of critical importance in many fields. The use of brackets (instead of, for example, commas, parentheses, or format on a page) is purely an idiosyncrasy of Logo, but the structure of that list is not. When students appreciate why the address is, itself, subclassified into street and city/state information, and apply that reasoning to the organization of data bases of their own design, they have learned something essential about heirarchical structure.

Because so much of mathematics involves transformations from one representation of given information (e.g., 2(x+7)=9) to another (x=⁻2.5), the projects so far have emphasized the creation of processors that perform the transformations and sources that house the representations. An important mathematical tool that has not yet been investigated is the displaying of information, itself just another form of representation. In Logo, the job of creating a display is handled by destination procedures like SETPOS, PRINT, FD, and SHOW. For special purposes, like graphing, student may need to create their own specialized destinations.

Let us create a "database" of vertices of a figure and display its contents graphically. Processors may be designed to reflect, translate, rotate, expand, or otherwise transform that figure. No new techniques—in particular, no recursion—are needed, and so attention can be focussed on the mathematics, not distracted away to the programming. Here is a part of such a system of procedures. SETPOS, rather than PRINT is used for displaying a point. A new destinations, DRAWTRI is needed for displaying a triple of points.

```
TO TRI1
 OP [[-80 -30] [-20 40] [20 -10]]
END

TO X :PT                  TO Y :PT
 OP ITEM 1 :PT             OP ITEM 2 :PT
END                       END

TO NEGX :PT
 OP LIST (PRODUCT -1 X :PT) (Y :PT)
END
```

```
TO DRAWTRI :PTS
 PU SETPOS ITEM 3 :PTS PD
 SETPOS ITEM 1 :PTS
 SETPOS ITEM 2 :PTS
 SETPOS ITEM 3 :PTS
END

TO DRAWNEGXTRI :PTS
 PU SETPOS NEGX ITEM 3 :PTS PD
 SETPOS NEGX ITEM 1 :PTS
 SETPOS NEGX ITEM 2 :PTS
 SETPOS NEGX ITEM 3 :PTS
END
```

The instructions DRAWTRI TRI1 and DRAWNEGXTRI TRI1 show how parts of the system work. The new destinations DRAWTRI and DRAWNEGXTRI, use techniques already seen in the data base and usable at top level. The processors X and Y are homologous to NUMR and DENR in the fraction calculator, and NEGX itself uses no techniques not already encountered in the fraction calculator.

The somewhat cumbersome form of the new destinations and the even sillier need that there be two of them is an attempt at keeping the programming "simple" and familiar, avoiding the need for students to learn recursion or complex constructions involving RUN before embarking on this project. A Logo augmented with the iteration tools MAP and MAP.LIST (see Brian Harvey's "Iteration in Logo" in the Logo 84 Pre-Proceedings) would allow DRAWTRI to be defined

```
TO DRAWTRI :PTS
 PU SETPOS ITEM 3 :PTS PD
 MAP [SETPOS] :PTS
END
```

It would also eliminate the need for a whole new command (DRAWNEGXTRI) to do what DRAWTRI did, but for a transformed set of points. Just as points could be transformed one at a time by NEGX (e.g., NEGX ITEM 1 TRI1), all the points in TRI1 could be transformed at once by MAP.LIST [NEGX] TRI1. Thus, DRAWNEGXTRI TRI1 could be replaced by DRAWTRI MAP.LIST [NEGX] TRI1.

My initially stated purpose was to concern myself in this paper—and focus my students—on the mathematics and not on the programming, so what justifies this latest excursion into alternative programming styles? Precisely my original goal! My intention to keep the programming out of the way of whatever other intellectual pursuit I and my students are engaged in obliges me to consider which programming styles will best serve present and future goals—which styles are most learnable and which most serviceable.

In this paper, I suggest several projects, styles of teaching, programming metaphors, and even novel iterative primitives. Experience so far suggests that, with careful selection of programming models and thoughtful attention to metaphor and meaning, it is not as difficult for students to learn functional notation and programming as is often claimed, but with all these variables changing at once, the question of what combination makes the most learnable package remains open for further study.

The question of what is most serviceable, however, is not an empirical question, but a theoretical one. The concept of functions is positively central to mathematics, and it is a shame to ignore such an important theme in our programming with students.

The Turtle Grows Up:
Using Turtle Graphics to Enhance Words and Lists Programming

MARK FRYDENBERG
Graduate Student
College of Computer Science
Northeastern University
Boston, Mass. 02115

Abstract: Many text-processing applications naturally lend themselves to the use of recursion in the Logo environment. The presenter will demonstrate examples of such projects from three different disciplines, will suggest techniques for adding graphic enhancements to text-processing procedures, and will share strategies for creating "tool procedures" to supplement Logo primitives. The higher-level thinking and problem solving skills needed for their implementation will also be developed.

Key Words: Recursion, Words and Lists, Graphic Enhancement, Tool Procedures, Problem Solving, Higher-Level Thinking Skills

This presentation is for experienced Logo users who have come to appreciate both the fun and aesthetics of turtle graphics, the useful structure of words and lists, and the power of recursion, and are interested in using Logo to develop higher level problem solving skills by incorporating all of these techniques in a set of Logo procedures.

The presenter has developed a series of advanced Logo applications from the areas of mathematics, computer science, and language which show how Logo's two main features, graphics and words and lists, may effectively be combined.

An Application from Mathematics: The Euclidean Algorithm

The Euclidean Algorithm is used to determine the greatest common divisor of two integers, and is a "good" programming exercise since its solution involves writing a recursive Logo procedure. By itself, however, the procedure gives little insight into what the greatest common divisor really means, but a graphics enhancement leads to new discovery and easier understanding of mathematical relationships.

An Application from Computer Science: A Recursive Descent Scanner

The presenter has developed an implementation of a recursive descent scanner for a simple right linear grammar using Logo. Turtle graphics are used to draw a finite state automaton and highlight transitions which display its current state as the automaton changes. While the procedures that "drive" the model are entirely text-oriented and recursive, adding graphics aids the user in grasping the concept of a scanner, and in turn, facilitates an understanding of how the main procedures "work."

An Application Involving Language Skills: A Popular Word Game

The presenter will explain the design process used to develop a complex set of procedures to play a word game, sharing problem solving strategies and solutions to some unexpected implementation issues. "Tool procedures" which accomplish specific programming tasks such as looping and printing sprite representations will be developed. Programming techniques including the use of modular design, recursive list-processing procedures, and integrating sprites in the graphics portion of the game will be discussed.

MICREL : A microworld for the social sciences.

by
Claire Fournier and Jacques Lafeuille*

Collège de Chicoutimi
Chicoutimi, Québec
and
APO QUÉBEC
Centre Québecois de recherche sur les applications pédagogiques de l'ordinateur
Montréal, Québec

MICREL is a MICroworld of RELationships, created for the social science. The concept of microworlds and their applications were described in some detail by Papert in Mindstorms. The idea of creating a microworld for the social sciences was sparked by Mindstorms, and fueled by our desire to create an environment in which students could be more in control of their own learning.

In order to construct a new microworld, it was first necessary to establish the requirements for microworlds in general, and the social sciences in particular. What new primitives would be necessary for a coherent, internally-consistent world ?The necessary (although not necessarily sufficient) properties or mother structures of these domains needed to be identified in order to make possible logical and relevant manipulations.

We concentrated specifically on political science, sociology, economics, and anthropology. From our personal investigations and Delphi-technique research conducted with subject matter experts, the following mother structures were identified : for economics, association and exchange; for political science, association and influence; for anthropology, association and codes; and for sociology, association and form.

Once the mother structures were identified, the next step involved the creation of primitives to exchange, influence, code, and create forms. Approximately thirty primitives were required. Instead of a turtle, the user addresses one or more "marbles", identified by user-determined names. Each marble is defined as having certain properties, and the characteristics of each are apparent on the screen. The different textures serve to indicate compatibility between "marbles" for certain relationships. Each "marble" may be attracted, repelled, give part of itself to another marble, or become part of a group to create new forms.

The number and variety of relationships which can be manipulated by the user is, therefore, extremely large.

Our presentation will detail the processes of first defining a microworld and secondly, creating a microworld in the social sciences. We will present the primitives involved in this microworld. Our view of MICREL is that it is a tool with a wide range of applications; these, as well as future research involving MICREL, will be discussed.

MICREL exists currently as a prototype commercial product. It was programmed in Logo (Microsoft) by LCSI in Montreal and runs on a Macintosh.

* Mr Lafeuille was also at *Collège de Chicoutimi* during the development of Micrel.

ProLogo: A Prolog-like Language Written in Logo

Christopher Wasser

As a term project for a computer science course at the State University of New York at Plattsburgh, I have developed and implemented a Prolog-like system called ProLogo. Prolog has been chosen as the core programming language of the Japanese Fifth Generation Computer Project (JFGCP), and while ProLogo does not provide all of the features of Prolog it does allow the use of rule-based programming, the major focus of logic programming. It was written in ST-Logo for the Atari 520ST personal computer, which is based on Digital Research's DR LOGO, so the code could be transported to any other Logo system with only minor modifications.

This system can be used at several levels of involvement:

1) as a monolithic language for experimenting with artificial intelligence and logic programming
2) as a toolkit of routines which provide Prolog-like features to Logo
3) as a starting point for research into advanced areas of language design and implementation

The place of Prolog in the classroom is currently being investigated in Britain [3]. At this time the results are encouraging for Prolog advocates, indicating that students find Prolog to be a useful learning tool. However, there is one drawback to Prolog gaining widespread use in the classroom: the high cost of a Prolog system and the hardware to run it. ProLogo, on the other hand, can be implemented on even the most inexpensive of microcomputers, allowing students and faculty to experiment with logic programming on a minimal budget.

The monolithic system is complete, providing both the ProLogo kernel and a command processor for creating and maintaining data. This allows inexperienced computer users easy access to the language for experimentation, much like the SOLO langaue developed at Open University in England [1,2]. ProLogo allows at least the same degree of flexibility as does SOLO, which makes it a good candidate for many of the same types of research (semantic network representations of human memory), as well as many which are not conductive to SOLO's approach.

This system could be used at almost any level of schooling, from grammar school, as in the British studies mentioned above, through high school and on into college, although at a college level it would probably be more useful to non-computer science students.

By using the system as a toolkit of Logo routines which provide automatic processing and rule-based decision-making, the user can take more complete control of the system. ProLogo processing techniques could be applied to problems in artificial intelligence. By combining elements of both languages, through facilities provided in the ProLogo toolkit, problems such as the "Hungry Turtle" [4] can be handle through ProLogo rules and data, while still providing graphic output by means of Logo's turtle. In the hungry turtle problem a program must be designed which will allow the on-screen turtle to locate another object, his food dish, through an obstacle-course. ProLogo provides the structures for building general purpose rules for locating the target and passing instructions for moving the on-screen turtle to Logo routines.

ProLogo could also be used as part of an advanced computer science course, with projects ranging from the use and critique of logic programming systems to the finding of alternative, and perhaps more efficient or flexible, methods of query resolution. The internal data structures could be examined and modified (nested lists, parallel lists, lists used as arrays, and even streams of lists are used internally), or extentions, such as a better data-base storage scheme of structures for modularizing code, could be implemented.

College level computer science courses could find ProLogo an easy-to-use, easy-to-modify logic programming language. Most versions of Prolog available of microcomputers do not include source code, and those which do are very expensive.

The ability to use ProLogo's nonprocedural structure along side Logo's procedural structure presents itself as a formidable experimental programming system. The relationship between the two languages is mutually beneficial: Logo provides list handling, graphic output commands and mathematical constructs which ProLogo does not have, while ProLogo provides the power of rule-based decision-making, data-base manipulation and goal-oriented evaluation which Logo does not have.

[1] Denenberg, S.A. (1958), "A Service Project for an Introductory Artificial Intelligence Course: Implementing SOLO in LOGO", *SIGCSE Bulletin*, *17*

[2] Eisenstadt, Marc, "A User-Friendly Software Environment for the Novice Programmer", CACM, Vol. 26, Number 12, December, 1983, pp. 1058-1064

[3] Ennals, Richard, *Beginning micro-PROLGO*, Harper & Row Publishers, New York, Second Revised Edition, 1984, pp. 14-21, 117-157

[4] Martin, Kathleen and Riordon, Tim, "Tessellation Revisited and Hungry Turtle", The Computing Teacher, May 1983, pp. 57-59

Christopher Wasser
Department of Computer Science
State University of New York
Plattsburgh, NY 12901

A New Tool for building and editing Discrimination Nets

Ruben Wegman

LOGO CENTER NEDERLAND

Almost every LOGO-user will have met the program called "ANIMAL" on his utility-disk. This is a classical example of a discrimination net. Allthough the program enables the user to walk through a database and also to extend its tree-structure, its facilities are very limited. This paper describes a new and more powerful system for building and editing discrimination nets.

The ANIMAL-program

When the starting procedure of the ANIMAL-program is called, it will start with the root of the "data-tree", printing thè question of the node and waiting for a yes/no-answer. Based on this answer one of the two subtrees is selected and the root of this subtree is used to repeat this sequence of actions. When the program finally reaches a leaf of the data-tree it outputs the conclusion which is stored in this leaf and asks whether this conclusion is right. If it is wrong, it will be necessary to modify the data-tree in such a way that the next time the conclusion will be correct. This can be achieved by changing the leaf into a node with two leaves as sons, one leaf containing the old, wrong conclusion while in the other leaf the new, correct conclusion is stored. Finally, it will be necessary to store a question in the new node (the old leaf) which discriminates between the two conclusions.

Limitations

The user of ANIMAL can build up a data-tree in only one way; just by following a path through the tree, starting at its root, and adding a new node and a new leaf at the end of the path. This procedure implies that when a small data-tree grows into a larger one, the path leading from the root to a leaf also grows in length. As a consequence, it takes a lot of time to construct a large data-tree. An even more important disadvantage of building discrimination nets this way is, that it is not possible to correct errors which have been made while building the tree. A major handicap of the program finally is that it has no facility for graphical display of tree-structures.

In view of these limitations, the user shall want to have a more powerful set of tools for editing discrimination networks. They should enable him to insert a new node in the data-tree at any possible place, to delete a node, modify a node, change subtrees, combine subtrees, etc. The discrimination net-user may also want to have a facility for graphically displaying the data-tree on his computerscreen or his printer in a trausparent way. But, what the user will need most of all, is the possibility to shorten the path from the root to a leaf. This can be achieved by permitting him to attach more than two possible answers to a node. In this way the program eliminates other conclusions faster than before, so that it will reach its conclusion in much less time. The second way of shortening the path is by making use of tree-transformations. As a result the data-tree remains balanced and the program also reaches its conclusion more efficiently.

A new tool for building discrimination nets

After considering these needs of discrimination net-users, the research group of the Dutch LOGO Foundation designed and implemented a prototype of a *LOGO discrimination net-builder* on the Apple Macintosh. This prototype includes all the features described

above and makes full use of the facilities of the Macintosh such as dialog-boxes, multiple windows, customizable pulldown-menus, etc.
Discrimination nets are frequently used in problem-solving tasks, where situations have to be classified on the basis of a large number of properties. With our prototype of a discrimination net-builder one could, for instance, make a discrimination net which traces malfunctions of cars by asking the user appropriate questions. Having diagnosed the fault, the program could suggest what to do to repair it. Or one could make a data-tree that enables a program to advise persons where to go on their holidays, by systematically asking them the questions which are encoded in the tree. Sample discrimination-nets built by the prototype demonstrate its efficiency.

For further information:

R. Wegman
Res. Dept. LCN
PO Box 1408
6501 BK Nijmegen
The Netherlands

KIDS AS SOFTWARE DESIGNERS:
An assessment of children's concepts of learning and teaching

A Proposal for The Third International Logo Conference, July 1986

By: IDIT HAREL
Arts and Media Technology Lab
Massachusetts Institute of Technology

Presentation Format: Using videotapes (VHS), an IBM PCjr, and an overhead projector, I will illustrate the opportunities offered to fifth grade programmers using all capabilities of Logo (text, sound, and graphics) when they are engaged in learning a subject through software-design or programming projects intended to teach younger kids.

Presentation Content: Logo as a tool for learning about learning and teaching. During the past year, the fifth grade children at the Hennigan School were involved in designing programs for second graders to teach fractions through Turtle graphics and text, or how to read through self-created animated story books. They were also encouraged to create manuals or "travel guides" for their diskettes' directories, so any child or adult can read through the directory and get an overview of what each program is doing, what can be learned from it, whether it is interesting, how to use it, etc.

At the Hennigan School (Project Headlight), Logo is being used as a tool for learning any subject matter (i.e., literature, motion, reading, math, music, writing, social studies, and much more). More emphasis is put on the use of Logo as a tool for combining domains (i.e., math with writing, music with motion). The computer activities at the Hennigan School are very much project-oriented. The children--who are using computers five days a week, for one hour and a half a day--do not learn Logo for the sake of learning Logo, but rather in the process of being involved in a project within a larger context.

Therefore, at the Hennigan School, the children have the opportunities to use Logo as a tool to explore relevant issues of instructional software design and implementation. Also for the researcher it is an opportunity to gather more information about the children's attitudes towards their programs, their theories about learning and teaching, and their cognitive styles.

One assumption of the presenter is that children's understanding of a subject matter and of programming itself can increase when they are involved in designing a piece of software to teach something to others. Their meta-thinking about learning and teaching, and their decisions throughout the design process, function as a window to what they see as a target of difficulty in a domain, and an intriguing environment to assess their theories about learning and teaching. In addition, the children's process of evaluating their own completed programs is essential for their understanding of how data flows within their programs, and whether the program achieves its purpose.

Moreover, during the process of developing Logo programs to teach others, the children have a chance to explore their own strategies and discuss new ones, regarding design issues such as: the "user-friendliness" of their own programs; the content and length of feedbacks; the balance between text and graphics on each screen with relation to comprehension and aesthetics; the pace of their programs; how much control they give to the user; the differences between on-line and off-line documentation; the need for menus; etc.

The presentation will include a demonstration of software written by the fifth graders and selected anecdotes will illustrate their process of making choices with relation to the above learning, teaching, and design issues. In addition, since the information gathered when looking at the children as software designers or curriculum developers provides another lens into my research on cognitive and learning styles, the dichotomies of design strategies or styles that can be correlated with the child's learning, personality, or cognitive state, will be presented as well.

Finally, after describing the pilot study of these issues which was conducted during the past year, I will present my plan for the Hennigan School's "Learning Technologies Design Workshop." The goal of the workshop (which will start this coming September) is to provide children with interactive videodisc systems, computers, slide-projectors, cameras, Logo, existing discs (as a source for visuals), posters, and crayons, as tools to develop their own materials. The workshop will allow children to study a subject through designing a lesson for teaching others. The workshop will also foster discussions among children on different presentations or design strategies that can illuminate the deep structure of a given subject matter, as well as discussions about their thoughts on learning and different teaching theories.

Presentation Target Audience: (1) teachers and educators who are looking for new ideas for projects, and related themes (i.e., instructional design strategies, children thinking about teaching, etc.); and (2) researchers who want to get new insights on how to broaden their lenses (or create new paradigms) when researching Logo as a tool for children to think about learning and teaching, and Logo as a tool to assess children's cognitive and learning styles.

Anne McDougall
Faculty of Education, Monash University
Clayton, Victoria 3168, Australia

CHILDREN'S DIFFICULTIES IN PERCEIVING STRUCTURE AND USING SUBPROCEDURES

Ideas of procedurality, modularity and problem structure appear to be difficult for many children to grasp and use with power. In this paper I shall look briefly at several sources of such difficulties which have been suggested by other researchers, and report some data on perception of design structure which might add to our understanding of children's problems in this area.

The most common ways of introducing children to procedures are probably having the turtle 'remember' the commands for a picture or pattern so it can be generated again later, and to avoid re-typing the commands to be used later. Hillel (1985) draws attention to learners' resulting identification of procedures with the objects they describe, so that children look on them as end products rather than in functional terms, as descriptions of processes. He suggests this non-dynamic view of procedures as one explanation for the persistence of interface bugs.

Leron (1985) looks in some depth at the problem of interface bugs. He points out that "When conceiving of a given picture as a hierarchy of subpictures and interfaces, the subpictures can be directly perceived in the original picture, whereas the interfaces cannot." (Leron, 1985, p.29). He sees this lack of any clear concept of the interface between subprocedures, and the subsequent lack of attention to the related importance of the turtle's state before and after each subprocedure, as reasons for the frequent difficulties children have with interface situations.

As well as problems such as a product-related view of procedures and difficulties with interfaces between procedures, evidence is beginning to appear that some children have difficulties associated with their perceptions of the design or drawing being made. Noss (1985) reports a tendency, particularly with younger children, for children to draw around the outside of pictures. For example, children asked to draw a "tower", composed of rectangles of decreasing size stacked one on top of another, wanted to draw the outline first, even though they had just built several different sized rectangles. Thinking that the problem was with the interfacing between the rectangles, the researcher asked them to consider drawing "steps", in which the rectangles were aligned at one end. Again they proposed an outline strategy, and only when presented with the task of drawing "blocks", four equal stacked rectangles, did the children agree (and then reluctantly) to use their rectangle procedure. Noss comments that their earlier experience with a slightly inaccurate floor turtle might have predisposed them against strategies that required "going over lines", but since similar difficulties were observed with separated shapes, he thought that "the difficulty seems much more related to perception rather than to execution" (Noss, 1985, p.183). Noss observed that the tendency for some children to draw outlines persisted even after they had had experience with procedures and had been introduced to the idea of sub-procedures.

Another case of a child regarding a "whole" figure as a basic unit is clearly documented by Lawler (1985). The child was working with a POLY procedure, essentially REPEAT 24 [FD 200 RT 90 WAIT 30].

> "When she described the object created as 'a square' and the activity as 'going around', I proposed we count the iterations to see if the turtle did it twenty-four times. I identified the unit of the turtle's action as the procedure step, but Miriam identified the unit as the making of a square. When we counted its actions, I counted out loud to '6' - that is one and a half squares. Miriam continued counting under her breath and concluded with '10'.... Miriam saw the thing made primarily as a 'square' and not as a shape emerging from the repetition of more primitive actions." (Lawler, 1985, p.172)

One of the children with whom I have been working has shown an approach to subprocedure use which is strongly affected by her tendency to see "whole" designs, concentrating on outlines. Nine years of age at the time she did the work reported here, this child was generally competent at writing procedures and in the syntactically correct use of sub-procedures. However at times she found difficulty in seeing the smaller component parts of a design, and as a result patterns which might have been easily executed became extraordinarily difficult for her. I shall illustrate with some examples from one of her Logo sessions.

Her first task, working from a reference book (McDougall et al., 1982), was to "draw three equilateral triangles arrayed around the turtle's central position". A diagram was shown in the book. A triangle procedure had been written previously, and the book gave a "Hint: turn the turtle 120 degrees after each of the triangles". The child did not at first seem to see the pattern as essentially symmetrical about its center, but described it as having two pointed "fronts", facing to one side. She commented that her shape's fronts, which were generated from right-turning triangles, would face in the opposite direction from the ones in the book as it used left-turning triangles. I wonder whether, without the hint in the book, she might have used translation, preserving the "fronts", instead of rotation; however she drew the pattern using the triangle procedure and turns of RT 120.

The next task suggested was "Draw a hexagon made up of six equilateral triangles, like this", and a diagram was given. Instead of taking one of the several "obvious" strategies based on rotation of the original triangle, she defined a new triangle, a left-turning one, and then rotated it to the left to create a mirror image of the first three-triangle shape. Her viewing the three-triangle figure as the unit led to a strategy using reflection rather than pure rotation.

Next the book suggested "Spin some squares about the central position to make an interesting design like these examples", and two patterns, one of eight and one of sixteen squares symmetrically arrayed about the screen center, were shown. The child had no idea where to start or what to do to generate these designs. She saw no relationship between these and the last drawings. She could see squares in the outer regions of the first pattern, but could not work out how to generate the "spokes" effect in the middle. I played turtle to illustrate rotation of a small square through 90 degrees about its corner at the screen center. She responded that that would make a "window", and wrote a procedure, called SP.SQ (special square) to draw that.

```
     TO SP.SQ
     REPEAT 4 [SQ LT 90]
     END
```

She then raised the pen, moved the turtle to the edge of the drawing, and could not continue. When I directed her attention back to rotation about the center, she returned the turtle there and rotated the window through 45 degrees about its center to achieve the first of the two patterns in the book.

She described the final pattern as "like that one, only with more of them ... like if you had a whole lot of square coasters you could sort of make it, but you wouldn't have the middley bit." This image certainly suggests focussing on the outlines, the window frames. After another fruitless pen-up trip to the outline, she achieved the pattern by again rotating the SP.SQ window about its center.

Noss suggests that children concentrate on outlines because they want to draw with the turtle as they did with a pencil - to draw the outline first. He conjectures further that a description of his "tower" shape, for example, by its outline and some internal detail lines, avoids inserting the extra "duplicated" lines which would be needed for it to be seen as a set of stacked rectangles. He also notes some children's difficulties in perceiving the invariance of a procedure, that its orientation is only dependent on the initial heading of the turtle.

Discussing the "triangle fronts" and "windows" examples with colleagues led me to another possible partial explanation - the role of language in perceiving design structure. Very early in her Logo work the same child had made a picture of an engine, which she spontaneously broke up into sub-pictures for the cabin, wheels, smokestack and so on, all of which parts have descriptive names. The square and triangle to be rotated in the textbook exercises, although named shapes, were less descriptive of the drawing being attempted than were her units for manipulation, a shape with pointed fronts and a window. This conjecture seems even stronger in Lawler's iteration example, described earlier, where the square which Miriam saw as the unit of the turtle's action could be named, while the procedure step itself could not.

I have attempted to draw attention to and exemplify difficulties that some children have in perceiving modular structures in designs, and the effect these difficulties can have on the children's ability to use subprocedures in their programming. Logo teachers (and textbook writers!) cannot assume that tasks seen by them as having obvious solution strategies will be viewed by students in the same way. A student's perception of a drawing might imply a strategy which is quite different, and possibly much more difficult.

ACKNOWLEDGEMENTS

I am grateful for Kirsty's contribution to the work reported in this paper. I have gained insights into parts of this material from discussions with Sylvia Weir, Sophie McCormick and Tony Adams.

REFERENCES

Hillel, J. "Some Thoughts on Logo Dichotomies", paper prepared for the World Logo Conference, 1985.

Lawler, R. *Computer Experience and Cognitive Development*, Ellis Horwood, 1985.

Leron, U. "Logo Today: Vision and Reality", *The Computing Teacher*, Feb. 1985, p.26-32.

McDougall, A., Adams, T. and Adams, P. *Learning Logo on the Apple II*, Prentice-Hall, 1982.

Noss, R. *Creating a Mathematical Environment Through Programming: A Study of Young Children Learning Logo*, University of London Institute of Education, 1985.

Weir, S. "Logo as an Empirical Window", in Sorkin, R. (ed.) *Pre-Proceedings of the 1984 National Logo Conference*, Massachusetts Institute of Technology, 1984, p.63-75.

A Further Development in the Theory of Learning Microworlds

Tony Adams
Royal Melbourne Institute of Technology
Melbourne, Australia

Anne McDougall
Monash University
Melbourne, Australia

Little thought appears to have been given in previous research to providing a methodology that encourages the idea of designing coherent sets of microworlds. The methodology discussed here is only a first step to providing a rigorous approach to microworld design. It is an overview of one aspect of microworld design, rather than a definitive description. Nevertheless it embodies a number of important principles.

The methodology shown employs the idea of a network of microworlds connected by required entry and exit skills. A single microworld may be decomposed into other microworlds that are completely contained within it. These in turn can be decomposed into other microworlds until some fundamental microworld is reached that cannot be further decomposed. This notion is vital because of the nature of the development of turtle geometry.

Turtle geometry has been previously shown by Groen (1984) to be a microworld. In a real sense, any student or teacher is likely to be dealing with a very small portion of turtle geometry at a particular time, or even in their entire experience. Lawler's polyspi microworld (Lawler, 1984) is perhaps the definitive and best reported example of a very constrained microworld within turtle geometry. Polyspi in turn leads to the inspi microworld and so on. Neither polyspi or inspi are capable of further decomposition Both can be composed into higher level microworlds by using their respective primitive operations within higher level primitives.

The only way to account for both turtle geometry and polyspi to be considered as microworlds is to provide for top down decomposition. This view intuitively does not appear to be in conflict with any previous work on microworld theory. The methodology shown is broadly based on work by Gane and Sarsen and DeMarco in developing a data driven approach to systems design. The methodology as described in its present state of development makes no claims as to the design of individual microworlds, nor of their use.

A microworld is shown as figure 1.

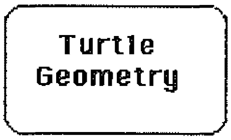


Figure 1

Each microworld will have a minimum of one entry and exit skill. These are shown in figure 2

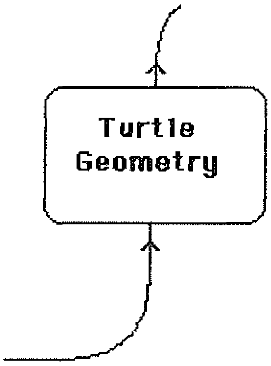


Figure 2

Microworlds can be joined together by entry and exit skills (figure 3)

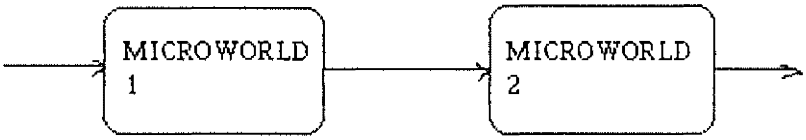


Figure 3

A skill may be developed in a non-microworld environment. This non-microworld environment, for example a series of muscle building exercises in Fischer's skiing microworld (Fischer, 1981) is shown as figure 4.

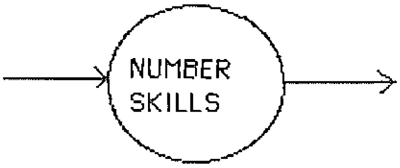


Figure 4

The decomposition of microworlds into other microworlds (and non-microworld activities) is shown in figure 5.

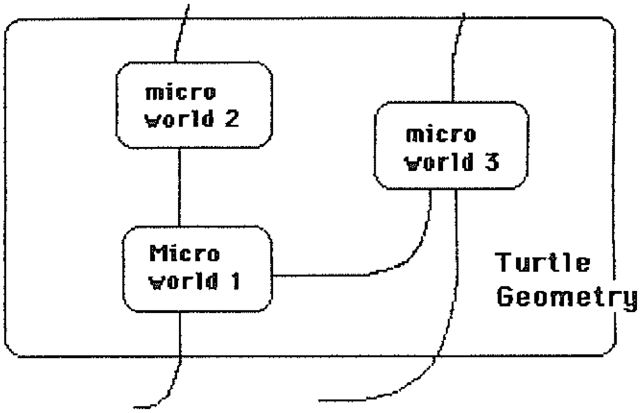


Figure 5

The skills leading into and out of the turtle geometry microworld are entry and exit skills for this microworld, as well as for component microworlds.

References

Fischer, G. "Computational Models of Skill Acquisition Processes" in Lewis, R. and Tagg, D. (eds.) Computers in Education, North-Holland, Amsterdam, 1981, p.477-481.

Groen, G, "Theories of Logo", in Sorkin, R. (ed.), Logo 84, Pre-Proceedings of the National Logo Conference, Massachusetts Institute of Technology, Cambridge, Mass., June 1984, p.49-54.

Lawler, R. "Designing Computer Based Microworlds" in Yazdani, M (ed.) New Horizons in Educational Computing,, Ellis Horwood, Chichester, 1984, p.40-53.

Honing Your Logo Skills:
Telling the Story of Logo Syntax

Laurence J. Davidson

Bolt, Beranek, and Newman, Inc.

There are many ways to tell the story of Logo syntax. What's useful for a novice isn't useful for an experienced programmer; what's useful for a teacher may not be useful for a student. Whether you're a teacher or not, this talk is addressed to you if you are an intermediate-level Logo programmer, especially if you've discovered that your model of Logo syntax works most of the time but has a tendency to break down in hard cases.

This paper is an abstract of a far longer one that will be handed out at my talk. The paper describes the full story of Logo syntax, a story which is not simple, but which (I hope) is complete and correct.

When you were a beginning and intermediate-level Logo programmer, your teachers helped you build up a model of Logo syntax through concrete examples and hands-on practice; this model was probably serviceable but certainly incomplete and almost certainly inaccurate. There's nothing wrong with that. As Saki pointed out in one of his short stories, "sometimes a little inaccuracy saves tons of explanation." It's the right way to begin. But eventually there comes a-time for complete explanation.

Will you benefit from a fresh way of looking at things? One way to tell is to check how readily you can provide appropriate answers to the following questions.

(1) Chris wants to define an `UNTIL` procedure to implement instructions like "Keep moving the turtle forward until a key is pressed" and claims that it must be invoked with a form like

```
UNTIL [KEYP] [FD 1]
```

rather than

```
UNTIL KEYP [FD 1] .
```

Dale, on the other hand, points out that the primitive IF uses a form like the second one above, so UNTIL should also. Isn't IF similar to UNTIL? Chris and Dale want to know who's right, and why.

(2) Is Dana right to reject the following instruction, on the grounds that it doesn't follow proper Logo syntax?

```
REPEAT :FOO :BAR
```

(3) Explain the difference between brackets and parentheses.

(4) You overheard Sandy pointing out to Lee that the first input to MAKE always has quotes before it. Lee disagrees; what is your comment?

(5) What will be printed if we type

```
MAKE "X 5
PRINT DOUBLE INCREASE
```

assuming that DOUBLE and INCREASE are defined as follows? Explain your answer.

```
TO DOUBLE :NUM
MAKE "X 2 * :X
PRINT :X
END

TO INCREASE
MAKE "X :X + 1
OP :X
END
```

Actually, the very phrasing of some of the questions above suggests inappropriate models of Logo, in ways that I will explore in my presentation. Some of them are even the wrong questions to be asking.

I have two bases for this claim.

One basis is that the questions focus too much on surface punctuation rather than on meaning. This is the major cause of breakdown of models of Logo syntax in hard cases. **If you find yourself recalling (or inventing) arbitrary rules about all those brackets, colons,**

parentheses, and quotes, then you need a fresh view, a new way of looking at things.

The second basis is that the questions implicitly ignore the regularities of Logo syntax. Punctuation marks and procedures don't follow *ad hoc* grammatical rules; they all mean something. Furthermore, each punctuation mark and each structural component in a Logo expression has a single consistent meaning. Once you really understand what brackets do, you won't find yourself muttering incantations like "Use brackets after IF in Apple Logo and after REPEAT in all Logos," or "Put quotes before the first input to MAKE." You won't have to memorize a lot of specific rules that not only are easy to get wrong but also block off alternative ways of doing things—ways that are often useful and sometimes necessary. And you won't find yourself giving incorrect explanations to students.

Incantations may be appropriate for BASIC, Pascal, and C, but they're not appropriate for Logo.

Logo has one consistent syntax. To understand it fully, you need to learn four sets of concepts: Logo phrase structure, evaluation, punctuation, and exceptions. My talk will touch on all four areas; my extended paper describes each in detail.

After hearing my talk, reading the paper associated with it, and reflecting on its contents, you should be able to fashion appropriate explanations for novices and students. The explanations may be the same as mine, or they may be different—but in any case I hope that you will understand mine well enough to come up with correct versions yourself. The full story of Logo syntax should help you clarify the structure of Logo in your own mind and should therefore let you articulate and explain pieces of it in ways appropriate for the particular audiences that you find yourself addressing.

By the way, Chris and Lee are right; Dana, Dale, and Sandy are wrong.

Proposal for presentation
LOGO 86

Creating Tools in Logo

Brian Silverman
Michael Tempel

Last year we presented a statistics microworld. We called it Fuzzy Logo. Among other things it had a turtle that didn't quite work right and useful primitives for playing with random numbers. We included primitives did things like compute some combinations and permutations, give an average of a list of numbers, or give a number that was "nearly" the same as another number.

These new "primitives" weren't and aren't part of the Apple Logo // system diskette that we used. They were, instead, written in Logo. There really isn't any reason to distinguish between a primitive and a procedure that you write yourself. In fact, Logo doesn't really see that difference. The only real differences are that a primitives tends to run faster (we hope) and that they can't be printed out. Other than that something that you write yourself can be used in exactly the same way as a primitive.

This session will be about techniques and styles for writing new "primitives". We've decided to call these tools. We hope to show that tools are easy to write and use. Learning how to "speak" Logo is somewhat like learning any other language, natural or artificial. Above all it requires practice. Learning Logo, however, is probably much easier that learning French. There are a lot fewer rules. In fact there are so few rules that people often get confused by trying to add some to explain away some of the syntax. Logo, like most programming languages has a quite straightforward but formal structure.

The session will present about two dozens examples of tools written in Logo. We hope that the examples can be generalized so that their scope extends to most of the problems that you will encounter. There really aren't that many different "idioms" to understand before being able to write most any procedure that you'd need.

A Logo procedure can be a command or a reporter. Reporters are sometimes called operations in some of the older literature. Commands are more familiar. These are procedures that tell Logo to do something. FORWARD, RIGHT and SQUARE are all examples of commands. Reporters report an answer. SUM and HEADING are reporters.

For example:

```
PR HEADING
```

asks Logo to print the turtle's current heading. HEADING reports the current heading and that is then used as the input to PRINT.

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

is a reporter that picks an item randomly from a list.

Most introductory Logo literature talk only about commands like SQUARE and SPINSQUARE and the like. To balance that off some we will concentrate a fair amount on operations. Recursion will also be covered, both as a programming technique and as a problem solving skill.

The contents of the session derive somewhat from a workshop we gave in St. Paul where we discussed tools in Logo.

For more information contact:
Michael Tempel
Logo Computer Systems Inc.
555 West 57th Street Suite 1236
New York, New York 10019

EXPLORING FRACTIONS WITH LOGO
(An example of using Logo as a tool to teach mathematics)

"Come see this, I just divided it into four parts."

"Four eights and one half are equal, I never knew that!"

"Wow, three eights is smaller than one half."

"If I multiply two fractions, they grow shorter."

"This can't be right. One fourth of one half can't be one fourth, can it?"

These comments are from students studying fractions in classrooms at the Carroll School. Each student has created a unique model which in turn is used to study simple fractions. Although different in size and color, all the models are similar in shape - they are all rectangular and can be divided into equal sections. Each model bears a different name, but are often refered to as "giant inches". Using these models the students are able to visually observe the effects of adding, subtracting and multiplying fractions. Logo, facilitates not only the construction of the models, but the effortless manipulation of the parts of the model as well.

I will incorporate the following in the presentation:

1) Construction of the model
2) Questions asked to stimulate exploration of concepts
3) Manipulation of the model

I will use student procedures developed with IBM Logo to illustrate my presentation. I have chosen to focus on one math topic in order to cover one sequence in its entirity. From this example, participants should be able to generate ways to use Logo to teach other topics.

The population of students at the Carroll School are "learning disabled". I have found Logo to be a particularly good tool for this population, but imagine it could be used in the same way with all types of students learning mathematics.

Logo and Fractions: A Case Study from the Hennigan School

Marlene Kliman
MIT Media Technology Lab
20 Ames Street
Cambridge, MA 02139

Can we use Logo to supplement and/or supplant traditional elementary school instruction in fractions?

How can Logo activities affect understanding of computations and underlying concepts related to fractions?

What sorts of Logo activities can be used for exploring fractions?

How do teachers feel about using Logo in this way?

A working group on Logo and mathematics at the Hennigan School, the site of MIT's Project Headlight, has been addressing some of these issues. The group, composed of Hennigan teachers and MIT Project Headlight members, explores ways of using Logo to meet teachers' needs. One emphasis has been fractions. Over the course of the year, we have developed and used specific kinds of Logo activities and projects which stress underlying conceptual issues related to fractions. The talk will focus on these activites, projects, and concepts: both actual content and relationship to traditional math class objectives and teaching methods. Handouts including sample programs and tool proceudres will be provided.

Many of the activities and projects center on creating, subdividing, and manipulating areas drawn with Logo turtle graphics. Subdivisions can be filled in or otherwise marked to represent fractional relationships. Working with Logo in this way can give concrete meaning to fractions, operations on them, and equations involving them, as children learn by actually constructing and manipulating. This personal involvement can lead to a deeper understanding of the meaning of fractional relationships, in particular, of traditionally difficult concepts such as division of fractions.

Specific activities related to fractions and Logo turtle graphics include:

- Creating multiple representations of a given fraction, varying factors such as shape, size, orientation, and number and type of subdivisions

- Repeating a unit module (perhaps a procedure) a given number of times with REPEAT

- Focusing on fractions as relative numbers of turtle steps composing subdivisions, rather than relative numbers of subdivisions

- Exploring equivalent fractions vs. equivalent areas
- Considering fractional relationships in perimeter, area, and volume
- Using superposition of shapes to explore common multiples

Other ways to explore fractions and operations on them with Logo include writing procedures that exchange quantities of money for their equivalents in different denominations, using Logo to create and manipulate meters, and using Logo to create and explore objects that move at different speeds.

EXPLORING INTEGERS WITH LOGO

BY ROBERT WINKLER
SHAWNEE MISSION SCHOOLS
OVERLAND PARK, KANSAS

Are you positively negative about integers? Do you have difficulty finding a good model for the addition and subtraction of positive and negative numbers? Do you have to resort to having your students memorize rules (without understanding them) in order for them to compute accurately? If your answer is yes to any of the previous questions, I have some ideas that will help you and your students.

One of the problems we have faced in teaching our students computation with integers is the lack of a good model to illustrate the process. This is not to say that none are available. However, most of them fail to meet one of the following criteria:

1) it correctly illustrates the processes involved
2) it uses objects or ideas to which students can easily relate
3) it is easy to learn and use
4) most importantly, it is easy to remember

Many teachers, therefore, teach their students a rule or "shortcut." But this method usually fails to meet the fourth point of the above guidelines. Students have trouble correctly remembering (and applying) rules that they do not understand.

This is where the turtle comes to the rescue! It can help teachers and students with more than just the geometry in their math curriculum. LOGO can be used to provide a wonderful model of positive and negative numbers that meets the qualifications stated earlier. I have some activities and ideas that are being used in our district to teach students how to compute with integers. Not only are they learning to add and subtract positive and negative numbers, but they are gaining an intuitive feeling for the process. Students use LOGO to help them develop their own rules for computation.

The concepts and ideas are quite simple, yet they are very powerful. Students only need a knowledge of the commands FD, BK, RT, LT, and HOME in order to participate. These activities may be used to help teach specific objectives in the math curriculum, or they may serve as a starting point for student explorations.

The presentation will use LOGO to cover the following topics:

1) the concept of a negative number
2) absolute value
3) developing an intuitive feeling for the addition and subtraction of integers
4) designing a model that students may use to help them the with the addition and subtraction of integers
5) multiplication with integers

I believe that LOGO will soon be seen as the natural and logical way to introduce integers to our youngsters.

For further information, please contact:

Robert Winkler
8641 E 97th Terrace.
Kansas City, Missouri 64134

Two Logo Staff Development Projects

by

Michael Tempel, Mike Hopkins,
Sharon Burrowes, Brian Silverman

The St. Paul Logo Project and the ECCO Logo Project are two long term staff development efforts that can provide a useful model for educators whose aim is to institutionalize Logo in their schools. The two projects are different in their origins and structures, but certain similarities have emerged that may be generalized to other situations. Both projects rely on a stable leadership group to foster and support the expansion of Logo activity. Outside experts work with the leaders who in turn provide training and support for a much larger number of educators.

We will report on the history and development of these two projects, the format and content of staff development activities, and the results of these efforts.

The St. Paul Logo Project

In November 1982 Logo was introduced into to seven schools in the St. Paul Community - School Collaborative. The Collaborartive is committed to changing schools and recognizes the need for those active in each school community - teachers, parents and the principal - to be involved in the change process.

Logo was presented as more than a computer language. It served as an introduction to the use of the computer as a tool for learning, to developmental theory and the philosophy of learning through discovery, and to learning cultures and environments where the focus is more on learning than on teaching.

Since 1982, the Logo Project has expanded to include 23 elementary and secondary schools. Over 3ØØ people, including members of the school board, parents, teachers and principals have been trained in Logo.

The initial training session was conducted by Seymour Papert. Throughout 1983, consultants from LCSI provided periodic workshops for the original group of teachers.

A major two-part workshop, which included several educators from outside St. Paul, was conducted for a week in April and a second week in July of 1984. Since then, two-day workshops have been held three times a year for the leadership group. An intensive one-week session is being planned for the summer of 1986.

Throughout this period, the leadership group has conducted introductory and intermediate training for an ever expanding group of teachers. This core group consists mostly of teachers, but also includes a school social worker, two parents and two high school students.

Several teachers act as Logo resource people for their own schools, providing information and support for their colleagues. One teacher now works full time as a Logo coordinator and the two parents work part time as Logo consultants, available to anyone who reqests assistance. Recently formed topic groups allow interested teachers to meet with resource people to develop activities which bridge from Logo to curriculum areas including math, art, science, social studies, pre-school and special education.

The ECCO Logo Project

ECCO, the Educational Computer Consortium of Ohio, an organization that grew out of a "teacher center", began its Logo training as early as 1979-8Ø when it presented a number of "What is Logo" sessions. It then became involved in a program known as "Catch on to Computers", sponsored by General Foods which used Logo on Texas Instruments Computers as a vehicle for learning about computers. Shortly thereafter, ECCO offered "Logo Discovery for Families" and "Logo to Go". The "Logo to Go" series provided participants with a Radio Shack Color computer and "Logo" for the duration of the workshop. In addition, ECCO's annual fair has included a Logo strand for at least the past four years. These introductory sessions, workshops, and presentations, were always on a beginning level, introducing participants, primarily teachers, to turtle graphics and procedure writing.

LCSI's first involvement with ECCO came in the spring of 1985 with a more advanced workshop conducted by Michael Tempel. Later that spring, with the help of funding from the Jennings Foundation, ECCO sponsored a highly successful Logo Fair which included sessions that were a direct result of this earlier workshop.

During the 1985-86 school year, ECCO has sponsored a series of workshops, partly funded by Jennings Foundation. These workshops have included introductory sessions, intermediate level sessions which focused on using Logo in the major curriculum areas, and a series of more advanced workshops presented by LCSI. The intermediate workshops contained material developed by participants in the advanced workshops. This material is, in turn, being used in classrooms and to train yet other teachers in a number of school systems in Northeastern Ohio.

In the spring, ECCO again sponsored a Logo Fair. This fair offered sessions and workshops at all levels for teachers from school districts in a rather large geographical area.

Staff Development Workshops

In both projects, introductory sessions begin with turtle graphics and are designed to get people involved with, and comfortable using Logo. Discussions of curriculum and Logo philosophy are also part of these sessions.

The two part workshop in St. Paul during 1984 provided sufficient time for people to become invovled with extensive projects. These efforts each focused on a particular curriculum area.

Since then, the one-day or two-day advanced workshops in both St. Paul and Ohio have each concentrated on a specific topic. These topics fall into two general categories - technical knowledge of Logo and connections between Logo and the curriculum.

Some of the curriculum workshops provide an overview of various possible Logo activities that relate to a subject area. "Exploring Langauge with Logo" included work on sentence and poetry generators, conversation programs, text editing, branching stories, language interpreters and pluralization and conjugation programs. Other sessions explore possibilities that emerge from a single starting point. "Fuzzy Logo" uses a slightly innacurate turtle to inspire explorations of statistics and feedback mechanisms.

While conducting these curriculum workshops we identify certain areas of technical Logo knowledge that need attention. During "Fuzzy Logo" in St. Paul, for example, we found that many people weren't comfortable with writing their own operations, especially recursive operations. The next session "Creating a Logo Tool Box" worked on these skills.

It seems difficult to effectivley emphasize both a content area and specific Logo skills at the same time. Lacking

Logo skills may interfere with the exploration of a subject area. Practicing Logo skills is best done in a familiar context.

Results

How can we measure the results of our efforts? The amount of Logo use in both projects has expanded. Increasing numbers of teachers and students are using Logo.

We also know that the level of technical skill has been increasing among people in the leadership groups. Comparing projects produced in recent workshops with those from earlier sessions reveals increasing complexity and more sophisticated uses of Logo.

It is not as clear that we have produced changes in the the day to day classroom environment. Perhaps the most difficult piece to achieve is changing the structure and culture of learning environments. Many of the schools which have been using Logo for several years have made sincere attempts to apply the philosophy of Logo. None has given up the curriculum. Instead, they are attempting to integrate Logo and their curriculm. Shifting the focus from teaching to learning is more difficult to achieve than acquisition of technical Logo skills.

It has become clear that in order to connect Logo to the standard curriculum, a teacher must have a thorough understanding of the concepts the curriculum hopes to achieve, sufficient technical Logo skills, and the vision to see connections when they present themselves. Finding "powerful ideas" and providing appropriate guidance to enable and encourage children to explore these ideas has proven most challenging. At LOGO 85 Seymour Papert cautioned the Logo community about "technocentrism". We should be clear that people, not technologies are responsible for change. The task before us now is helping people to make changes.

For more information contact:

Michael Tempel
Logo Computer Systems Inc.
555 West 57th Street, suite 1236
New York, New York 10019

Mike Hopkins
St. Paul Public Schools
360 Colborne Street
St. Paul, Minnesota 55102

Sharon Burrowes
Wooster High School
Wooster, Ohio 44691

Brian Silverman
Logo Computer Systems Inc.
9960 Cote de Liesse
Lachine, Quebec H8T 1A1

SKILL, EXPLORE, PROJECT: A SUCCESFUL TEACHER SUPPORT PROGRAM

David Chesebrough
Sewickley Academy

KEYWORDS: Implemention: Teacher training; Teacher/Student Support materials; Classroom/Individual; Integration; Lower/Middle Grades

This presentation will address the issue of how to support the underprepared teacher starting Logo while still holding true to the Logo environment and moving the teacher towards independently supporting successful student experiences.

PRESENT LOGO DILEMMA

The Logo "revolution" has been succesful enough to progress beyond a relatively few inspired, imaginative teachers who held true to the spirit of a "pure" Logo environment of exploration and minimal (but timely and skillful) teacher guidance.

Now, however, Logo has been picked up (in many cases forced) into the hands of "the masses" of teachers who have little idea of the nature and nuture of Logo. Districts make administrative decisions and mandate that Logo be taught, and then provide a 2 day workshop. It's like telling every teacher they will start teaching French and then offering a weekend introduction in the langauage prior to the first class!! And we wonder why Logo is bogging down in some cases, or being dropped altogether?

In the ideal Logo environment a skillful teacher spots the "teachable moment" where the student can be introduced to a new concept and encouraged to explore at a new and exciting level. However, many classroom teachers now using Logo have neither the language background, the general computer skills, nor the philosophical underpinnings to direct the students properly without guidance. In many situations the students are set in front a computer with Logo and instructed to explore, pretty much on their own. However, it has been observed by many and echoed in several well known articles that the resulting aimless wandering through Logo results in stagnation and loss of interest.

ONE SOLUTION

I had to deal with the problems of supporting an administrative decision in 1983 that every teacher in grades 3 - 6 in our school provide the Logo instruction for their classes . In response, comprehensive teacher and student support materials were developed which organize learning around fundamental skill levels and then guide students (and teacher) through each level in an exploratory environment.

In this session, I will explain and demonstrate the use of the various parts of the support system of Logo materials we have developed, which include:

Teacher background material explaining concepts and offering classroom analogies and off computer ideas;

Activities to introduce and develop the various fundamental skills, and encourage exploring with the skills;

Extensions which either offer avenues for further exploring, or cover optional concepts and/or skills. (Every level has extensions supporting it which offer challenge shapes, project ideas for integration into the curricular topics, and teasers which jump the student ahead to get a taste of higher level skills and greater power of Logo).

Projects which draw upon previous skill levels, and can serve as jump off points for the next level of skills.

Troubleshooting advice and potential problems to avoid.

Technicalities of the computer (disk use, printing, etc);

Teacher Support Disk of procedures, microworlds, and sample programs.

As an anxious teacher progresses using the support materials, the feeling of accomplishment breeds more comfort with Logo which translates into a more relaxed, exploratory environment - the one we Logo veterans find at the heart of its offering. In turn, the positive student response encourages the teacher to allow even more flexible and freer Logo use, which the materials are designed to support as well.

Sample activities and an outline will be distributed. The entire support kit will be published by a major publisher and information about obtaining the complete set of materials will be available.

For further information, please contact:

David Chesebrough
Computer Coordinator
Sewickley Academy
Academy Avenue
Sewickley, Pa 15143

PROPOSAL FOR PRESENTATION AT LOGO 86

SUBMITTED BY:
Ihor Charischak
Logo Computer Systems Inc.
555 W. 57th Street (Suite 1236)
New York, NY 10019
212 765 4780

TOPIC: Living the Mindstorms Vision: A Model for Training the Teacher Trainers

> "..Computers (used in a particular way) can be powerful carriers of powerful ideas and of the seeds of cultural change...they can help people form new relationships with knowledge that cut across the traditional lines that separate humanities from the sciences and knowledge of the self from both of these. (Mindstorms, p. 4)

Seymour Papert's vision for education has inspired many people, including myself, to go out and share our interpretation of it with friends and colleagues. Not only have we been sharing something that was personally powerful, but we were also making available to people the possibility of a more humane educational system. Now, several years later, many of us continue to bring the message to people, but clearly the honeymoon is over. We now have to answer the questions of the skeptics who challenge the viability of the Logo approach. One way that we defend Logo is by pointing at inadequate teacher training. But what does constitute good teacher training? My presentation is an attempt to answer this question.

In thinking about what experiences teachers of Logo need, I looked at Papert's vision statement for a clue and was impressed with two important ideas. The first is that computers need to be used in a particular way if they are to be a source of empowerment and a carrier of powerful ideas. I realized that one of the reasons that Logo has gained such popularity in the schools is because teachers have discovered that this unique approach to computers is not only personally fun to do, but also has value for children. The second idea is that Logo can be a seed for cultural change. In almost all beginner workshops there is some discussion of logo philosophy, its Piagetian roots, and the characteristics of the "logo environment". But there is rarely a discussion of cultural change. This is understandable, since teachers for the most part come to these workshops to get some "hands on" experience with Logo and many of them associate cultural change with utopian dreams which may be interesting to talk about but not very doable in their classrooms.

Another reason why the Logo culture is struggling to grow is because it is trying to survive in a world of more established, conflicting, and even contradictary cultures. For example, the logo culture seeks to assimilate a softer view of computer use (Turkle, The Second Self, Simon & Schuster, 1984) within a culture of male dominated computer users who have a "top down" view of how computers should be used. Many computer educators who are products of this more established culture see nothing wrong with imposing their style of learning on children who may learn differently saying that it is for their own good.

Turtle geometry is clearly getting logo into many classrooms. But it is only bringing with it half of the Mindstorms vision. If we want the ideals of Logo to be resonant in the schools, then the trainers of Logo teachers must help them to operate as if the vision was actually possible.

John Naisbitt, in his book, "Re-inventing the Corporation", says that if individuals can envision and articulate a future they want, they can more easily achieve their goal. Vision is a link between dream and action. To help teachers approach their teaching from this point of view requires that they first see the value of living from one's vision. Secondly, they must create an action plan to make it happen.

In order to appreciate the value of living ones vision, teachers must be able to think about their teaching. They should be able to step back and observe what they do and why they do it. They should notice how cultural biases come into play and they should resist blaming other people or institutions for problems.

Teachers of Logo teachers need to learn how to deal with issues that have an emotional "charge" associated with them. Here are some of them.

1. Logo criticism.

In cultures where there is a tradition of constructive criticism, educators should take research results in perspective realizing that the research models may not be appropriate for what they are trying to measure. The Logo educators should resist "backlash" reactions which only strengthen the position of the critic. The educator needs to think about what needs to be said and deliver it in a responsible manner.

2. Approaches to teaching Logo.

Papert asserts that the best kind of learning is the kind that occurs without teaching, sometimes called Piagetian learning. Uri Leron (Logo Today: Vision and Reality, The Computer Teacher, 2/85) believes that such learning is not realistic and proposes a quasi-Piagetian approach which encourages appropriate teacher intervention. The potential problem is that teachers can confuse a sensible teaching strategy (quasi-Piagetian) with vision. Clearly the goal of all teaching should be that students learn without our intervention. That doesn't mean that students should not follow our rules, tap our brains, or listen to our sometimes boring lectures. In other words, we should encourage Piagetian learning while we set up quasi-Piagetian strategies.

3. Evaluations

Schools are run by educators who must answer to the people who are responsible for the welfare of our schools. It is the foundation of our American educational culture. As long as there are formal institutions, there will be a need for evaluations. Rather than resist the use of evaluations, educators need to look for ways to assess what children are doing in ways that will empower students and at the same time meet the needs of the institution (a quasi-Piagetian approach to evaluation). This opens the possibility for new ways of evaluation to become a part of the "mainstream" educational culture.

A workshop for trainers of Logo teachers should include the following:

1. Discussion of what it means to teach

Some sources of information include the writing of Dan and Molly Watt (Teaching with Logo, Addison Wesley, 1985) who have taken a close look at what teachers actually do with Logo. Tom Peters (A Passion for Excellence, Random House, 1985) looks at what qualities characterize good leaders.

2. The role that learning styles plays

Sherry Turkle (The Second Self, 1984) describes poignantly the lives of children and the unique impact of computers on each of their lives. Teachers need to be sensitive to individual needs and concerns.

3. Teacher as learner

In schools Logo fosters the creation of subcultures of learners. This is consistent with the vision of logo: learning without teaching. The teacher needs to learn how to participate in these cultures.

4. Teacher as innovator

The teacher must come with new ideas to share. No matter what opinion students have of their teachers, they are always ready to respond openly to something new or interesting. Teachers need to discover and nuture their creative talents.

A Working Model for Training Teacher Trainers

The idea for this model grew out of an actual workshop that I conducted in New Jersey this past fall. The workshop consisted of two consecutive sessions that met on a weekly basis for six weeks. The first session (2 hours) was for educators who are interested in training teachers in using Logo. The second session (3 hours) was a training session for teachers who are interested in teaching Logo to children. The participants of the first session assisted me in the training of teachers in the second session. The agenda for the first meeting with the Logo trainers consisted of the topics above as well as a discussion of Logo content and strategies for implementation. Each week a different teacher from the first session is responsible for planning and doing the second session. On the following week during the first session, I would discuss and evaluate with the Logo trainers their work in the previous week's training session.

Two Logos: The Bridge is Recursion

Stewart A. Denenberg
Department of Computer Science
SUNY at Plattsburgh

At Logo 85 Uri Leron suggested the existence of two Logos: the Logo of elementary turtle graphics and the Logo of advanced list processing. He also suggested there was no smooth path between the two. He was right about the first part and wrong about the second. That there are two Logos is clear to anyone who has learned the language and especially to the hundreds of teachers every year who are promised workships on "Advanced" Logo (list processing) and are frustrated and disappointed to learn they still can't understand it.

I believe the critical concept which bridges the two Logos is the idea of recursion. Unfortunately, a bridge separates as well as spans two areas and so it is necessary not only to locate the bridge but to find a safe, comfortable and possibly even enjoyable way of making the journey. I propose that the transition between the Logos be made in two stages, progressing from graphic recursive procedures to simple statistical recursive functions which manipulate lists of numbers and return values. Parallel to this pedagogy is dynamic two-dimensional tree notation which can be used to represent the recursive process: the breadth of the tree represents each instruction in the program and the depth represents the recursive calls while dotted line branches show that recursion is just a process of problem reduction that uses the strategy of "hanging in there" until the reduced procedure is fully executed.

Stage 1: Graphic Recursive Procedures

We assume the learner has seen this "dumb" type of recursion to draw a box:

```
TO DUMB.BOX  :S
REPEAT  4  [ FD :S  RT  90 ]
DUMB.BOX  :S
END
```

It's dumb for three reasons:

1) It draws the same Box over and over itself when once would have been sufficient.
2) The procedure never stops - it has no provision for termination.
3) It makes beginners think recursion and iteration are the same.

A 'smarter' version of the Box procedure that addresses the first two criticisms is:

```
TO SMART.BOX  :S
IF  :S < 2   [STOP]
REPEAT  4   [ FD :S  RT 90 ]
SMART.BOX  :S/2
END
```

The following extensions to the Box procedure which are not simple tail recursion address the third criticism and provide the motivation for developing a notation that represents the recursive process:

```
TO  BOX1  :S
IF  :S < 2  [STOP]
REPEAT 4  [FD :S  RT 90]
BOX :S/2
LT 45
FD  :S
END

TO  BOX2  :S
IF  :S < 2  [STOP]
REPEAT 4  [FD :S  RT 90]
BOX :S * 0.8
REPEAT 4  [RT 90  FD :S]

TO BOX3  :S
IF :S < 2  [STOP]
REPEAT 4  [FD :S  RT 90]
BOX :S * 0.8
REPEAT 4  [RT 90  FD :S]
LT 30
FD :S
END
```

(BOX3 is a combination of BOX1 and BOX2 and produces interesting output when the argument is 32.)

We encourage the learner to enter and execute these three BOX procedures and the results are usually so surprising that no further motivation is needed to develop a representation of recursion so that procedures such as these can be traced and understood.

A Tree Diagram can be used as the notation for describing recursion - it is dynamic in nature and can easily be shown in the classroom on a chalkboard. Figure 1 is an attempt to represent the dynamic Tree Diagram for the execution of BOX1 in a static form.

In the Tree Diagram each instruction of BOX1 is shown across the breadth of the tree; the recursion occurs in the depth of the tree. The dotted lines represent "hanging" instructions to be executed after a recursive invocation is consummated and the numbers in the little circles represent the order in which the instructions are actually executed. It is also useful to show the screen after an instruction which changes its state.

Stage 2: Simple Statistical Recursive Functions

Once the learner has acquired an understanding of graphic recursive processes, the next step is to explain the concept of functions: procedures that return (OUTPUT) a value associated with their name. We can view these procedures as super-variables that assign themselves values (picture a variable that in addition to holding a value, also contains a procedure that produces that value). An easy way to begin is by showing the recursive process to sum a list of numbers where my capabilities have been reduced to being able to add only two numbers at a time. (I could compute the sum of 4, 3, 2 and 1 if only I knew the sum of 3, 2 and 1 because then all I'd have to do is add 4 to that sum and I'd be done, and I could compute the sum of 3, 2 and 1 if only . . .). Because SUM is already a Logo primitive we define:

```
TO SUMM:L
IF EMPTYP :L  [OP 0]
OP FIRST :L + SUMM BF:L
END
```

Using top-down design we can write a Correlation procedure in terms of a Standard Deviation procedure in terms of a Variance procedure in terms of a Mean procedure in terms of a Summ procedure in terms of a Length Procedure.

When we actually develop the recursive functions we alternate between top-down and bottom-up design:

1) We begin with the specification of the MEAN in terms of a SUMM and a LENGTH function which is yet to be written - we then redevelop SUMM in terms of LENGTH (which replaces EMPTYP) and finally write LENGTH wholly with primitives as shown in the listings in the Appendix.

2) We then take a small diversion and discuss how we would change MEAN so that it is more robust and handles division by zero and finally develop a shell program DRIVER to more rigorously test the MEAN procedure.

3) Next we define the Variance (VAR) in terms of the MEAN: The variance of a List of values X is the mean value of the differences (between each individual element in X and the mean of all X) squared. This necessitates the development of three ancillary procedures: DIFF which subtracts a constant value from each element in a List and SQ and PROD which multiply (dot product) List elements.

4) The Standard Deviation (STD) is then defined in terms of the Variance.

5) The Correlation Coefficiant (CORR) is defined in terms of STD, MEAN, DIFF and PROD as well as a new procedure DIV which divides each element in a List by a constant value.

6) Finally NEWDRIVER is a shell that tests out the total packages of statistical procedures.

In addition to teaching recursive list processing techniques, the statistical package illustrates that program development in practice can (and usually does) alternate between bottom-up and top-down technique in much the same way that an artist paints a picture. The listings of these procedures along with their driver programs are attached as an Appendix.

The advantage to this approach is most teachers have encountered educational statistics and so the ideas and their value to them is clear - certainly clearer and more valuable than using recursion to reverse lists of characters.

After concrete lists of <u>numbers</u> can be handled easily with recursive procedures, the user has safely crossed the bridge between the two LOGOs and can begin to look at more abstract applications that manipulate lists of words and finally lists of lists - a not overwhelming concept once recursion is mastered. Granted the crossing is not an easy one for most people, but if it can be made in stages where the first stage begins with a familiar model (Turtle Graphics) and the second stage uses concrete examples of Lists in familiar applications (elementary statistics), the journey can be made and be made smoothly by proceeding slowly and care-fully.

We have to constantly keep in mind an updated version of Hamming's admonition in his Numerical Analysis text, "The purpose of Computing is <u>insight</u> not Numbers [nor Pictures nor Strings nor Lists]". Recursion is not only a programming technique, it is a process that develops insight into what a procedure is and how it does it and, as such, provides the foundation for a bridge between the two Logos.

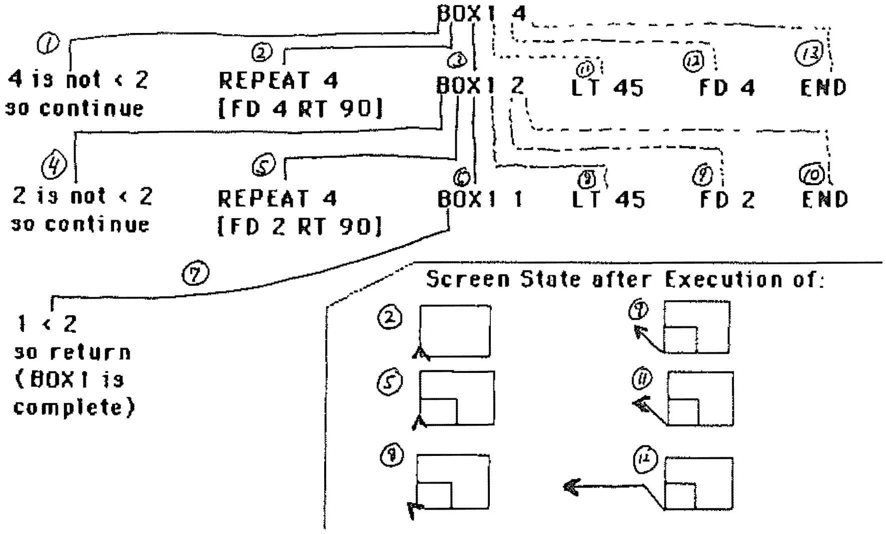


Figure 1

Tree Diagram representing recursive execution of:

```
TO  BOX1  :S
IF  :S < 2  [STOP]
 REPEAT 4  [FD :S  RT 90]
BOX :S/2
LT 45
FD  :S
END
```

APPENDIX

```
TO NEWDRIVER
PR [TYPE IN THE FIRST LIST:]
MAKE "X RL
IF EMPTYP :X [PR [NO NULL LISTS ALLOWED] STOP]
PR [TYPE IN THE SECOND LIST:]
MAKE "Y RL
IF EMPTYP :Y [PR [NO NULL LISTS ALLOWED] STOP]
IF NOT EQUALP LENGTH :X LENGTH :Y [PR [LISTS MUST BE OF EQUAL LENGTH] STOP]
PR []
PR SE [THE CORRELATION IS] CORR :X :Y
PR SE [THE FIRST MEAN IS] MEAN :X
PR SE [THE SECOND MEAN IS] MEAN :Y
PR SE [THE FIRST STDEV IS] STD :X
PR SE [THE SECOND STDEV IS] STD :Y
PR "  PR "
NEWDRIVER
END

TO CORR :L1 :L2
MAKE "ZX DIV (DIFF :L1 MEAN :L1) (STD :L1)
MAKE "ZY DIV (DIFF :L2 MEAN :L2) (STD :L2)
OP MEAN PROD :ZX :ZY
END

TO STD :L
OP SQRT VAR :L
END

TO DIV :L :C
IF LENGTH :L = 0 [OP []]
OP SE ((FIRST :L) / :C) DIV BF :L :C
END

TO VAR :L
OP MEAN SQ DIFF :L MEAN :L
END

TO PROD :L1 :L2
IF LENGTH :L1 = 0 [OP []]
OP SE ((FIRST :L1) * (FIRST :L2)) PROD BF :L1 BF :L2
END

TO SQ :L
OP PROD :L :L
END

TO DIFF :L :C
IF LENGTH :L = 0 [OP []]
OP SE ((FIRST :L) - :C) DIFF BF :L :C
END

TO DRIVER
PR [TYPE IN THE NUMBERS:]
MAKE "X RL
IF :X = [] [STOP]
PR SE [THE MEAN VALUE IS] MEAN :X
DRIVER
END

TO MEAN :L
OP (SUMM :L) / LENGTH :L
END

TO LENGTH :L
IF EMPTYP :L [OP 0] [OP 1 + LENGTH BF :L]
END
```

```
TO SUMM :L
IF LENGTH :L = 0 [OP 0]
OP (FIRST :L) + SUMM BF :L
END
```

Lists, Data, and Notation Systems

The first thing I liked about Logo was not its turtle, but its lists. My wife thinks lists are congenial to me because I am so messy that I appreciate anything that has a natural order. After all, it's absurdly easy to store information in a list - and to retrieve it you only have to locate its position in the list.

We start a Logo course in our school by having students build uncomplicated (but quite powerful) data bases. A simple phone directory is easy to create. Let a procedure **Directory** output a list of lists, each of which contains a list of names and phone numbers. Then write retrieval procedures. To accomplish this, you will probably discover that the data in the original lists should be re-structured.

Consider a simple phone number like 617 259-9527 which can straightforwardly be represented by the list [6 1 7 2 5 9 9 5 2 7]. This list has the advantage of simplicity of form and the disadvantage of simplicity of structure: its elements are neither grouped by category nor distinguished by function. The restructured list [[6 1 7] 2 5 9 9 5 2 7] contains the same numbers regrouped to promote retrieval of area code information. Other lists, like [[6 1 7] [2 5 9] [9 5 2 7]] and [[617] [[259] [9527]]], provide different groupings and hence different hierarchical structures, each of which reflects a particular organization of data.

Notation Systems as Data

Standard notation systems of arithmetic use lists to store numerical data. For instance the numeral 235 is essentially the list [2 3 5] in which the position of each element determines the value of that element: the 2 represents the number $2X10^2$, the 3 represents $3X10^1$, and the 5 represents $5X10^0$. The value of the list [2 3 5] is determined by adding the values of each data-element in the list.

Normally we don't need more data because numeration lists are understood to refer to a base 10 notation system. However in the context of several possible bases, the numeration list must include information about the base. There are at least two natural possiblilities in Logo. The simplest is to use the list [2 3 5 10]. This, however, both fails to differentiate the

10 from the other numerals and fails to group the digits of the numeral 235 to allow possible calculations. The Logo list [[2 3 5]10] reflects standard notation and supports algorithms that deal with notation lists.

Investigating and Applying Algorithms

I have had a lot of fun creating Logo procedures that operate on notation lists. Occasionally, I have re-invented an algorithm without realizing it, thereby (in a small way) really doing some mathematics. Furthermore, my experiments have allowed me to suggest mathematical projects to students, and I'm slowly getting better at letting them find their own successes without showing off mine.

An Evaluation Project

Most of the projects involving number bases are obvious. In particular, a project to convert numerals from one base to another is both an interesting challenge and a foundation for later algebraic projects. A good start is to write a procedure which recursively converts any numeral in a given base to a numeral base10.

Such a procedure must somehow reduce the original numeration list and call itself on the reduced list. Suppose, for instance, we are to evaluate the numeral [[3 2 5] 8]. To do this we must multiply 3 by the square of 8, add the product of 2 and 8, and finally add 5. We might begin by first multiplying 3 by 8 and then try to incorporate the result in a new and smaller list on which the same process might be called recursively.

Project 1

Write a procedure which recursively evaluates a notation list of any length to any base. Eventually consider bases greater than 10.

Once this is done, there are several related projects that cry for attention.

Project 2

Write a procedure which converts a numeral in one base to a numeral in any other base. Then try to improve the efficiency and elegance of the solution.

New Directions

A prodedure which evaluates numerals also evaluates polynomials. After all, the value of the polynomial $3x^2 + 2x + 5$ when $x = 8$ is the same as the value of the numeration list [[3 2 5] 8]. The standard numeration system for arithmetic is also a notation system for polynomials - with an evaluation algorithm already in place. Moreover, one can derive synthetic substitution and division, and the factor and remainder theorems by examining how the algorithm works on polynomial lists.

Project 3

Use the evaluation procedure for polynomials to write a procedure that outputs the quotient and remainder when any polynomial over the integers is divided by a linear factor.

There are many projects that can grow out of this one, among them projects to factor polynomials, solve polynomial equations that are factorable, and approximate the solutions to polynomial equations that aren't. My interests branched into projects involving standard operations with polynomials. If we can add and multiply polynomials, so can Logo.

Project 4

Write Logo procedures to add and multiply polynomials of any power. Once these are completed, write a procedure to exponentiate any polynomial to any non-negative integral power. Then use it to take a 10 th degree polynomial to the 10 th power in less than a minute!

Conclusions

In the presentation, I'll discuss solutions to some of the problems I have presented here, and suggest possible applications to mathematics education.

Jeff Sandys Logo 86

LIST GAMES

Young students quickly grasp turtle graphics because they have a first hand experience of going forward and turning right. List processing is more difficult to grasp because students do not have a physical experience of it. The games we will play during this session are designed to give students a "feel" for list processing.

Logo uses nodes to process list. Each node points at two other nodes. In these activities each student plays a node, pointing at two other nodes. By issuing instructions one player directs the construction of the lists. The students form the sentence and tree structured lists. And they get a node's experience of being searched, sorted and squashed.

The session will end with a short discussion of Logo's nodes and how students differ. A handout of instructions and related Logo procedures will be available. First hand experience makes learning easier. Mister Cons says be a good node and join us.

Wandering in a Sea Of Text

Mark J. Guzdial
Bell Communications Research
925 Church St., Apt. 4
Ann Arbor, MI 48104
(313) 995-5026

Introduction

We use reading and writing to communicate our thoughts to others and for explanation of another's thoughts, via memos, letters, articles, and books. But all of these forms of communication are shallow copies of human thoughts. Paper records thought linearly, while we think multi-dimensionally.

What would be far more useful would be a system where we could communicate our thoughts with a full depth of meaning. For example, it would be nice if when we wrote that a staff meeting is to occur at 1235 Baxter St., we might also be implying (for those who were interested) that 1235 Baxter is the large gray building next to the deli with the terrific pastrami sandwiches.

Allowing this system to convey greater depths of meaning can provide even more utility, especially in education. This sort of system can be the format for textbooks or class notes with multiple levels of detail. Consider what it would be like to read a piece of text on Valley Forge, and be informed that more information exists on George Washington. Continue your reading with George Washington, and be informed that additional information exists on Martha Washington, or on Presidents in general.

Perhaps you might want to use such a text tool for recording your own class notes as a student. As you learned new facts, you could enter more information into your system, drawing connections between subjects that you know are related and that you might want to investigate later. Imagine being a teacher reviewing your students' class notes recorded in such a manner. Your students' way of thinking about the material -- what was important, what was related, how it was related -- are there in front of you to review and use in gaging your students' understanding of the material.

I'm developing a computerized system in Logo to try to make possible these scenarios. The program is called REFTOOL because it's a TOOL for entering text and making REFerence links between subjects and files of text. A draft version has been completed on an Apple //c in LCSI Logo, and a second version is under development for use with other computers and other forms of Logo.

Using REFTOOL

Imagine that you have a disk with files of information on the Revolutionary War. Using REFTOOL, someone could set up "reference links" between, say, "Valley_Forge" and "G_Washington" (this means that REFTOOL has been told that Valley Forge refers to George Washington). There might be other reference links as well, perhaps between "Presidents" and "G_Washington," and maybe others as well.

Say that you then type in some information on Martha Washington in the file "M_Washington." You would want to tell REFTOOL that "G_Washington" should have a reference link to "M_Washington" (that makes the most sense as a link-- Martha wasn't a President, nor is she well-known for being involved in Valley Forge, but she certainly is linked with George). You would type

```
REFERS "G_WASHINGTON "M_WASHINGTON
```

Imagine that sometime later you're perusing your sea of Revolutionary War text, and you type in

```
PRINT REFERENCES "VALLEY_FORGE
```

and see the reference to G_Washington. Typing

```
SHOW WHOREFERENCES "G_WASHINGTON
```

would display

```
[VALLEY_FORGE PRESIDENTS]
```

and

```
SHOW REFERENCES "G_WASHINGTON
```

would display

```
[ M_WASHINGTON ]
```

So you can tell that you have more information about George Washington in a general sense in the file Presidents, or you could go on to read more detail about George and his life in the file on his wife, Martha.

You might not have wanted to go to such detail in finding information on George. REFTOOL lets you use more "brute force" methods by

```
POALLREFS "G_WASHINGTON
```

which prints out all references to George Washington throughout the library (or data base) of text.

Perhaps you're searching for information on the Potomac, but realize that you never created any references to the Potomac River. REFTOOL lets you do general searching throughout all the files on the disk by saying

```
FIND "POTOMAC KEYWORDS
```

REFTOOL also lets you group information on the text according to subjects. So your search for Potomac could be limited to a certain subject, say pre-1780 with a command like

```
FIND "POTOMAC :PRE1780
```

As you can see, REFTOOL permits grouping and linking of text, for later retrieval through the reference links, or more generally via keyword searches among all files or a subject-related subset of the files on the disk.

Implementation

As mentioned, each piece of text is stored in a text file on a disk with a keyword name. Each keyword known to REFTOOL is also a property list with two properties, REFERTO and REFERFROM. So G_Washington from the previous example, might have a property list that would look something like this

```
     PPROP "G_WASHINGTON "REFERTO [M_WASHINGTON]
     PPROP "G_WASHINGTON "REFERFROM [VALLEY_FORGE PRESIDENTS]
```

The heart of REFTOOL is the procedure REFERS. REFERS updates the reference links, updates the subject listings, and adds to the list of all keywords (for a new keyword).

The second version will have some changed command names, but will keep the same basic functionality with some additional features. The two main additions will be a browsing facility and a reference count associated with each keyword. The browsing facility will permit display of the textual data with its reference links (to and from), and the access of further data items without repeated execution of the core Logo procedures of REFTOOL. Also, associated with each keyword will be a property called DISPLAYED and an integer count. This count will reflect the number of times this text has been referenced by a user, a useful thing to know for many applications.

Applications

Many of the applications of REFTOOL were described in the Introduction section of this paper. One of my favorites is using REFTOOL to record students notes[1]. Such a notebook can lead to some interesting learning experiences. In a test of REFTOOL, I created a text database of quotes from Bartlett's Quotations. In a search for references from 1920, I found keywords for Mark Twain, Carl Sandburg, and interestingly, Mussolini. Using REFTOOL to point out links that already exist but might not yet be realized is a powerful idea.

Adventure games can be developed using REFTOOL. Each text file "visited" could represent another location or experience to explore. An adventure game like this could even be written by several people, where each participating writer might add to the story that's been told so far, or could go back and add new levels of meaning to sections that have already been described. Such a story could take on the depth and richness (and size!) of a tale like The Lord of the Rings.

REFTOOL has many other applications. It can be used for keeping track of literature searches. As described earlier, a text database containing calendar entries can be used to refer to files of additional information, for example, on who a meeting is with and what the topics for discussion are. I'll be taking an independent study this summer to explore other possible applications of REFTOOL, and for developing these that I've mentioned.

[1] An idea suggested by The Learning Tool, a tool for the Macintosh developed by Robert Kozma at the University of Michigan School of Education.

PROGRAMMING WITHOUT PROGRAMMING

by

Loren Abdulezer
Director, Management Services
Biller & Snyder, Certified Public Accountants
75 Maiden Lane
New York, NY 10038
(212) 425-5090

For many years people have been doing problem solving on computers without using computers as thinking tools. The general style or approach has been to create algorithmic routines that mechanically perform the steps dictated by people. Logo is a major departure from this philosophy. It demands that the user utilize the computer as a thinking tool to explore and formulate conceptions about the problem environment. Computing with thoughts about problems being solved is the crux of what Logo is all about.

To an extent, Logo provides a reasonably natural means to program through the use of definitions. For example, if we attempt to define a Factorial of :N as :N times Factorial of :N - 1 we might write:

```
TO FACTORIAL :N
IF EQUALP :N 0 [OUTPUT 1]
OUTPUT :N * FACTORIAL :N - 1
END
```

The above procedure illustrates that Logo has the flexibility to incorporate abstract definitions into the programming process. The fact is that programming in this style ignores other very powerful facilities which Logo is capable of providing.

Our firm has been applying Logo to business and financial problem solving. Our approach has been to carry Logo one step further -- problem solving does not per se, require programming (in the traditional sense). The following example uses a special primitive [**MAKEDEF**]we created in LM™ Logo (which runs on the Macintosh computer):

Constraint Propagation: an example

MAKEDEF is similar to MAKE in that a list can be assigned to a name. The list can contain values, procedures, instructions or other names created by MAKEDEF. Unlike MAKE, MAKEDEF utilizes what is called constraint propagation at toplevel. Constraints can be defined in any arbitrary order. This translates into the ability to create complex problem solving models without having to write programs. The following is an example of how this process works.

Imagine you have a retail store which sells shoes. You are going to introduce a new product line at the bargain price of $30 a pair. You can acquire these shoes at a (bargain) price of $19.50 a pair.

You would tell Logo:

```
MAKEDEF "SELLING.PRICE [30]
MAKEDEF "UNIT.COST [19.5]
```

So far MAKEDEF is like MAKE. Let us describe this problem a little further. In selling the shoes you are going to incurr two types of expenses -- fixed and variable. Fixed expenses could include such things as store rent, salesmen salaries, promotional ads and other miscellaneous expenses. Hence,

```
MAKEDEF "FIXED.EXPENSES [RENT + SALARIES + ADVERTISING + OTHER.FC]
```

The annual rent might be $60,000, total salaries for sales staff $200,000, $80,000 advertising budget, and other miscellaneous expenses of $20,000. You would enter the following:

```
MAKEDEF "RENT [60000]
MAKEDEF "SALARIES [200000]
MAKEDEF "ADVERTISING [80000]
MAKEDEF "OTHER.FC [20000]
```

The variable expenses are the costs that can be associated with each pair of shoes to be sold. Aside from the unit costs of $19.50 your store might have a policy of giving each salesperson a $1.50 commission for each pair of shoes they sell. You might also decide to hold in reserve a manager's commission. For the time being you are going to keep it to zero. You would write:

```
MAKEDEF "VARIABLE.EXPENSES  [UNIT.COST + COMMISSION]
MAKEDEF "COMMISSION [1.5 + MGR.COMMISSION]
MAKEDEF "MGR.COMMISSION [0]
```

For each pair of shoes to be sold you are going to clear a certain margin which is the difference between your selling price and the direct costs associated with each sale.

```
MAKEDEF "UNIT.CONTRIBUTION.MARGIN [SELLING.PRICE - VARIABLE.EXPENSES]
```

Your net income is going to be your total contribution margin less total fixed expenses. Your total contribution margin is equal to the contribution margin for each pair sold multiplied by the number of pairs sold. Hence,

```
MAKEDEF "NET.INCOME  [CONTRIBUTION.MARGIN - FIXED.EXPENSES]
MAKEDEF "CONTRIBUTION.MARGIN [VOLUME * UNIT.CONTRIBUTION.MARGIN]
```

Of course, the total sales revenue is selling price times the sales volume.

```
MAKEDEF "SALES [SELLING.PRICE * VOLUME]
```

At this point we can ask Logo to provide some answers. If we sold 35,000 pairs of shoes what would be our net income?

```
MAKEDEF "VOLUME [35000]
PRINT NET.INCOME
-45000
```

Then how many pairs would we have to sell to break even and what would be the total revenue? By definition break even is defined as net income of zero.

```
MAKEDEF "NET.INCOME [0]
```

The general definition of sales volume based on an anticipated net income is:

```
MAKEDEF "VOLUME  [(FIXED.EXPENSES + NET.INCOME)/(SELLING.PRICE - VARIABLE.EXPENSES)]
```

All we have to do is to tell Logo to give us the volume and sales.

```
PRINT VOLUME
40000
PRINT SALES
120000
```

The approach to problem solving using MAKEDEF and other constraint propagation tools is non-traditional to conventional programming languages and even to Logo itself. It fits comfortably into the entire framework and spirit for which Logo was originally intended.

Knowlege Representation In Logo: A Conceptualization Tool for Students

Steven Roffman, Ph.D.
Departments of Medicine
Columbia University
College of Physicians and Surgeons and
St. Luke's/Roosevelt Hospital Center
New York, N.Y. 10019

We are developing teaching methods that use computers and Logo to help students and researchers conceptualize the interrelationships that exist among many of the disciplines of medical science. We have implemented a knowledge-representation language in Logo, called LABFRAMES, which is capable of both representing descriptive information as well as relational knowledge.

Our underlying premise is that the process of representation of knowledge itself will aid in its comprehension and that it is possible to create powerful representation tools using Logo that will provide new heuristics to aid the learning of complex knowledge. While this work was initiated to describe knowlege in the fields of biochemistry and medicine, we believed from the outset that the use of a good representation system could help students of all ages conceptualize concepts in virtually any subject domain.

Suitable representations have played significant roles in how we think about complex ideas. Just as people need special representations to visualize the structures of our solar system and chemical molecules, suitable representations of relational knowledge, such as the causes of disease or the interdependencies of parts of the body, are needed to help us conceptualize and work with these kinds of knowledge.

In order to conceptualize large amounts of abstract information, it is often useful or necessary to to group complex knowledge into units. Minsky[1] suggested knowlege can be compartmentalized into units he called frames. A frame is a cluster of information about some object or abstract concept, represented as slots and values[2,3]. A frame for a specific person can be represented in Logo as the following list:

```
[Rachel [ISA [GIRL]]
        [FATHER [Steven]]
        [MOTHER [Peggy]]
        [BROTHER [Andrew]]
        [favorite.tv.show [The Cosby Show]]
]
```

In frame-jargon, the words ISA, MOTHER, FATHER are "slots," and the values of these slots are the lists following each slot.

Slots can be frames themselves. MOTHER can be a frame:

```
[MOTHER [ISA [PERSON]]
        [SEX [FEMALE]]
        [CHILDREN [one or more]]
        [SPOUSE [unspecified]]
]
```

PERSON, a value, can also be a frame:

```
[PERSON [ISA [HUMAN.BEING]]
        [NAME [unspecified]]
        [ADDRESS [unspecified]]
]
```

We can see that Peggy is a Mother (Rachel's mother). We can infer that she is a PERSON and that she is a human being by looking at the ISA slot of MOTHER and the ISA slot of the ISA slot of MOTHER (the ISA slot of PERSON).

While careful consideration must be given in the assigning frames and slots, the process can be beneficial in helping students analyze the knowledge they are trying to learn. The analysis can be done privately by each student, or in a classroom setting. The representation can begin simply, as a hierarchical list structure such as [AIR [OXYGEN NITROGEN CARBON.DIOXIDE]], expanding this to [AIR [ISA [MIXTURE]] [STATE [GAS]] [COLOR [COLORLESS]]]. Finally, using primitives of LABFRAMES, a set of conventions can be established for describing the kind of knowledge under consideration. Logo programs specific to the domain can be written to access information. To find chemical elements which are all gases the Logo program would create a list of elements whose STATE values were gasses. The user would write the procedures GAS:

```
TO GAS :FRAMES :GASES
IF EMPTYP :FRAMES [OP :GASES]
IF EQUALP GETVALUE FIRST :FRAMES "STATE "GAS
       [OP SE FIRST :FRAMES GAS BF :FRAMES :GASES]
OP GAS BF :FRAMES :GASES
END
```

(GETVALUE is a primitive of LABFRAMES which outputs the value of a given slot of a given frame).

The most interesting aspect of a frame-based knowledge representation system is the way information can be passed from one frame to another. The stereotypic knowlege about some general object can be automatically be assumed to hold for some specific instance of the object. The ease in specifying how inheritance can be performed is dependent on the choices in the design of the frame system. While there is much controversy on the best ways to implement inheritance, looking at the ISA slot is so convenient that it will be adequate for a great many representations of various domains of knowledge.

The common use of frames has been the creation of knowledge bases which permit programs to be written to perform human reasoning tasks such as medical diagnosis. These programs, called expert systems, rely on appropriate inheritance rules and representations of knowledge. The programmer, usually called a knowledge-engineer, works closely with human experts in some area, such as a physician or group of physicians, and translates the knowledge of the expert(s) into a computer representation. The process often results in the programmer becoming quite expert in the expert's domain.

It is the learning process involved in representing knowledge that we believe can significantly improve the ability of a student to conceptualize complex knowledge. Rather than create expert systems with knowledge bases, students can create personal knowledge bases which can be searched by them. As the student adds new knowledge to the knowledge base, along with new connections to existing frames, the student can use the system to uncover relationships that in the data that may be significant but not explicit.

A good frame representation of complex data will permit the computer to help a student in other ways. As the student encodes knowledge into a frame network, memory "frames," or memory schemata[4], may spontaneously stimulate the user to focus on other memories, and a process of refinement of the original thought may be initiated. This process of recall of past experience and its application to new problems is a key component of learning, and a computerized frame network will augment the memory of the user, and facilitate the learning process.

Logo is an ideal language for students to use with a frame-system. They can easily write or use common Logo programs which sort or find similarities between two lists, and can examine the knowledge base both through access programs such as GETVALUE, or by looking at list structures directly using the Logo editor. It is of great advantage to use the frame language while remaining in a normal programming environment. The only limitation is the memory of the Logo environment. Currently, LABFRAMES runs best on a 512K Macintosh, although it can be used for conceptual purposes on a 64K Apple IIe or IBM PC.

Our goals for LABFRAMES continues to be to develop a sufficiently powerful set of frame primitives to permit even novice Logo programmers to use the system, and to provide a set of Logo procedures to demonstrate how knowledge that the student has represented can be manipulated. Every student can then use the computer and Logo as a learning tool in a truly personal way.

The benefits we anticipate resulting from our studies with LABFRAMES is the enhancement of our understanding of how students learn and a heightened understanding of learning processes in general. This can then lead to a quantitative as well as qualitative increase in that learning.

References:

1. Minsky, M. A framework for representing Knowledge. M.I.T. AI Laboratory Memo 306 (1974) Also. In: *Psychology of Compueter Vision*, P.H. Winston, Ed. McGraw Hill (1975).

2. Winston, P.H. Learning by creating justifying and transfer frames. In: *Artificial Intelligence, an MIT perspective*, pp. 345-374. The MIT Press. 1979.

3. Rich, E. *Aritificial Inteligence*, pp.230-231. McGraw Hill, N.Y. 1983.

4. Boden, M. *Artificial Intelligence and Natural Man*, Basic Books, N.Y. 1977.

Logo and Social-emotional Development

The development of social competencies during the school years has relevance not only to overall social-emotional adjustment, but also to academic success and later participation in society. The influx of computers into schools has led to concerns about increasing social isolation; on the other hand, there are claims that computers are potential catalysts of social interaction. This paper will review qualitative and quantitative research concerning social-emotional development within Logo environments (there is space for only brief examples and synopses here, especially for the qualitative studies; the paper will include complete listings, descriptions, and references). Implications for the creation of Logo environments facilitative of social-emotional competencies will be drawn.

Interestingly, some observational evidence indicates that Logo may have its most potent influence in the area of social and emotional development. Relevant research will be discussed in terms of four fundamental aspects of social-emotional competence.

Social Initiation and Participation

Initiation and participation involve children actively seeking and maintaining interactions with the social and physical environment. Consistent with other instructional computing research (e.g., studies of CAI), there is evidence that the introduction of Logo environments does not interfere with social interactional patterns (Bowman, 1985). Fire Dog (1984) surveyed 29 teachers of over 600 students in grades 1-12. Teachers reported that children exposed to Logo programming were more likely to interact with peers. Research from Bank Street demonstrated that 8- to 11-year-old students tended to talk to each other more about their work when they were doing programming tasks than when they were doing noncomputer tasks (Hawkins et al., 1982). They did talk to each other when working on other classroom tasks (e.g., mathematics or language arts), but the subject of their conversations was often not related to what they were doing. Similarly, Kinzer et al. (1985) found that students working in Logo exhibit more learning-oriented interactions than do those in normal classrooms. Thus, Logo environments appear to have the potential to facilitate social interaction, as well as positively focus that interaction on learning.

Social Problem Solving

Social problem solving is the ability to effectively apply problem-solving skills to real-life situations, reflected in the ability to work and play cooperatively. Students in the Bank Street research engaged in more collaborative activity during computer than noncomputer tasks.

One study conducted with first and third graders using either Logo or CAI has indicated that children worked cooperatively more often on computers (with either

Logo or CAI) than off (Clements & Nastasi, 1985). Interestingly, they also got into more conflicts (possibly merely because they interacted more). However, children working with Logo, compared to children working with the CAI materials, were more likely to resolve conflicts. Exposure to CAI, in comparison, generated more oppositional and play behaviors in off-computer tasks, and more dependency behaviors. Opportunities to experience and resolve conflicts is necessary for the development of social problem-solving competencies. Therefore, Logo contexts may enhance the development of specific problem-solving skills.

Social Sensitivity

Social sensitivity is the awareness of others' feelings, a concern for their needs, and a willingness to share and help. Teachers have said that the greatest impact of computers in the classroom is that children tend to help each other more (Becker, 1983).

Hawkins et al. (1982) reported the computer context was the one where children more consistently identify certain of their peers as resources for help, indicating that Logo may facilitate the development of social sensitivity. In Logo, students learn to cooperate, listen, be critical in a constructive fashion, and appreciate the work of others (Burnett and Higginson, 1984). Fire Dog's teachers reported that Logo programming tended to increase teaching, consulting, and sharing in students. Clements and Nastasi (1985) found that childen working with Logo were more likely to help each other than children working with CAI. Although somewhat diffuse, these observations tend to indicate that Logo can positively influence social sensitivity.

Effectance Motivation

Effectance motivation is the degree to which children desire to control or effect change in the environment. It subsumes independent, self-directed work, internal locus of control, instrinsic motivation, attitudes toward learning (curiosity, enthusiasm), self-concept, and pleasure at intellectual discovery. Teachers report that students working with computers are more enthusiastic about learning, work independently more often, and take more pride in their work--students exposed to Logo are less bored in the classroom and exhibit more pleasure in their work (Fire Dog, 1984). There is a good deal of qualitative evidence that some students who were previously not committed to their school work became intellectually involved and more self-confident working in the Logo environment, often transferring these new attitudes into the classroom (e.g., Badger, 1983).

Milojkovic (1983) reported that 5th graders involved with Logo tended to take less responsibility for positive outcomes, but more for negative outcomes, a pattern characteristic of mastery-oriented students. They also placed a significantly greater value on a measure of independent judgment. This subscale produced the only significant difference on a measure of intrinsic motivation. Similarly, Schwartz et al. (1984) found no overall difference on a scale of intrinsic motivation; however, Logo students were more likely to use independent judgment. Brown and Rood (1984) found positive effects on interalized locus of control. These studies reveal an interesting pattern: Although generalized effects were not observed, students experiencing Logo did appear to judge situations for themselves and accept responsibility for their actions.

A large scale evaluation project revealed that fourth graders trained in Logo had slightly less anxiety toward mathematics, and more confidence in learning mathematics, than had control students (Scwartz et al., 1984). Similarly, Clarke (1985) reported significant increases in attitudes toward mathematics in girls experienced with Logo. Blumenthal (1985) reported no significant effects on low achievers' self-concepts, but significantly higher academic self-concepts in students who engaged in Logo programming. However, Lehrer (1985) reported no changes in attitudes within preschool children exposed to Logo, and Milojkovic (1983) found that computer groups scored significantly lower on one subscale of a measure of perceived worth. Inconsistent results would not be surprising, given that many students do not accept that Logo and mathematics are homologous enterprises (cf. Papert, 1980). However, the pattern appears to be a positive, albeit slight increase in attitudes toward mathematics, but little or no influence on general self-concept.

Clements and Nastasi (1985) found that children in a Logo environment exhibited on three behaviors indicative of self-efficacy: engagement in self-directed explorations, showing pleasure at discovery, and demonstrating the self-determination of rules. Long-term studies are required to determine whether such gains are consolidated, but these findings do provide evidence of Logo's power for enhancing effectance motivation.

Summary

It would appear that Logo--at the very least--has the *potential* to serve as a tool in encouraging prosocial interaction, social problem solving, social sensitivity, and effectance motivation (possible mitigating factors will be discussed). The social interactions that occur in Logo environments may be qualitatively different from those in other environments. In fact, child-child interactions during Logo programming may be as significant for cognitive development as are the child-computer interactions.

Douglas H. Clements
Kent State University
College of Education
401 White Hall
Kent, OH 44242

Logo, Robots, and Motivation

Scott R. Garrigan
Educational Technology Program
College of Education
Lehigh University
Bethlehem, PA 18015

The paper will link robotics literacy to traditional concepts of Logo programming. A successful method of teaching the use of the Logo editor will be presented, and a program will be described in whicb over 150 children used Logo and robots. Finally, the results of a pilot study will be presented which explores the relationship between Logo, robots, and motivation.

Educational Robotics

Educational robots have made their first appearances in schools over the past few years. Some schools now have a *robotics literacy* curriculum reminiscent of early computer literacy courses. The *robotics literacy* curriculum sometimes involves learning watered-down elements of formal college-level robotics courses. The author will present a case for robotics literacy being presented more like Logo and less like computer science or mechanics.

Over the past two years, the Educational Technology Program at Lehigh University's College of Education has investigated educational robotics. A graduate level study group was formed to pursue the investigation. A variety of robots and robotic activities were tried with children aged 7 to 14 who attended Lehigh's Summer Enrichment Program for the Mentally Gifted in 1984 and again in 1985. The robot activities were presented to the children much as Logo concepts are taught: by example, exploration with support, and trial-and-error with immediate feedback. A pilot study was conducted in the 1985 program to determine the relationship between student motivation and the particular robots and their features. The presentation will offer a description of the robots and the activities used, the teaching techniques, the pilot study, and the results of the pilot study.

Robot and Softbot Activities

The Logo turtle is the best known educational robot, but it is seldom considered as such. A floor turtle is an example of a robot, and a screen turtle is an example of a *softbot*. The author defines softbot as a screen robot which has no physical existence, but which responds to commands as would a physical robot. The youngest children were introduced to robotics through TRAINER, an INSTANT Logo program that is very easy to use. Actual Logo commands were used to guide the turtle through an assortment of onscreen maze programs which kept track of the number of moves. The maze program reported to the children when they had won, and how many moves they had taken to do so. Programming and use of the Logo editor were introduced as a means to make the turtle traverse a maze in one move. This technique has been successfully used by the author to teach the use of the procedure editor for two years, with about 180 children. He found that students are so highly motivated to reach the goal of winning with few moves that they are not frustrated by the many new concepts and skills necessary to learn to use the editor. The concept of a goal is very motivating to children.

Robot Odyssey and *Final Conflict* were other softbots used in the program. Each one requires children to program robots; one physically and the other with logo-like commands. *Robot Odyssey* offers rather full-featured softbots that can exercise control over their environment by picking things up, pushing or pulling activators, and by controlling other robots. The concept of control is also very motivating to children. *Final Conflict* allows children to program an army of robots, giving all of the directions in advance before the first one is released. Children can play against each other or against the computer in *Final Conflict*. *Final Conflict* offered the opportunity to study the effect of competition in learning to program robots. Both *Robot Odyssey* and *Final Conflict* evoke strong intrinsic fantasies in their players. These fantasies play a role in how motivated children are to continue playing the game.

The program used a variety of inexpensive physical robots. The *255 Computer Command Car* was a programmable mobile Corvette with high programmability and intrinsic fantasy. The *Armatron* was an arm robot with high control over its environment and low programmability. The *Horstbot* was a tank-like mobile robot with a claw-arm; it was interfaced to an Apple // microcomputer and controlled by logo commands and procedures. The *Horstbot* was high in control and high in programmability. The *Scorpion* was the most intelligent robot and the hardest for children to use. The author created menu-driven software to allow children to access its ability to move, play music, and to test its primitive vision system. The *Scorpion* was low in user-friendliness and high in novelty and capability. Pairs of children were given example tasks to perform with each robot that made use of the unique features of each.

Common Benefits of Logo and Robotics

There are many commonalities to learning Logo's turtlegraphics and learning how to program a robot. It is therefore possible that similar benefits can be expected if certain key commonalities are retained. In each case an object is commanded to perform a task by giving it a sequence of simple directions. In fact, Logo has been used as a robot control language since the first floor turtle. One technique for interfacing a robot to Logo involves a machine language driver that provides the interrupt capability to ensure two-way communication between the robot and Logo in the computer (Garrigan & Harvey, 1985). Programming a robot requires the same kind of problem solving approach as does Logo's turtlegraphics. We can expect children to exercise their problem solving skills, to plan ahead, to work cooperatively, and to work with accuracy, logic, and appropriate sequence. Even more closely tied to the Logo environment are the visualization and geometrical skills that are practiced while working with Logo's turtle or a physical robot. The presentor will make the case for a robotics literacy curriculum that is similar in spirit to a Logo curriculum.

Student Motivation, Logo, and Robots

Research has determined several factors that underlie the intrinsic motivation of children. These factors include goal orientation, skill mastery, self-esteem, curiousity, competition, creativity, and fantasy (Malone, 1980, and Csikszentmihalyi, 1975). These are the powerful motivators that toy companies, video game makers, and exellent parents and teachers use to capitivate the attention of children. It is possible to incorporate many of these powerful motivators into Logo application programs (such as the mazes discussed above) and into robotic activities (such as the robot Corvette). By tapping into the child's wellsprings of motivation we promote educational activities that are enjoyable, satisfying, and important to the child. The child will bring all the power of his native learning ability to such a task.

Pilot Study: Robots and Motivation

Robots today occupy the place in children's motivation that video games held in 1980. Children are captivated by robots. On any Saturday morning half of the cartoons involve robots, toy robots line department store shelves, and every science fiction movie now includes its share of robots. What is it about robots that turn cbildren on? Could it be the same factors that motivated children to spend hours playing video games? The author conducted a pilot study at Lehigh's 1985 Summer Enrichment Program for the Gifted to help formulate profitable avenues of inquiry about the relationship of robots to motivation. The priliminary study sought to identify the variables in a programmable robot system that affect child motivation. In simple terms, what makes an educational robot fun to use? The results of the study will be reported.

References

Csikszentmihalyi, M. *Beyond Boredom and Anxiety.* San Francisco:Jossey-Bass, 1975.

Garrigan, S. R. and Harvey, F. A. *Extending Logo's Power and Utility Using Machine Language.* Logo 85, MIT, July, 1985.

Malone, T. W. *What makes things fun to learn? A study of intrinsically motivating eomputer games.* PhD thesis, Stanford University, 1980.

Logo without the surrounding culture

Dr. A. J. (Sandy) Dawson
Façulty of Education
Simon Fraser University
Vancouver, B.C. V5A 1S6

In his call for papers for Logo '86, Brian Harvey invited reflective papers on three topics, one of which focused on the question, "Whatever happened to the revolution?"

This paper is one response to that question. It is based on my experience with the implementation of Logo compared to implementations whicb bave occurred over the years in a variety of curricular areas but centred primarily around mathematics.

The first point I wish to make is that in fact very few implementations of Logo have actually occurred on any large scale basis. It is true that some scbool districts bave incorporated Logo into their normal programs, but the scope of such endeavors is no where close to being of a magnitude that one could argue that the revolution has yet begun. As a result, I think we need to seriously consider whether or not the original question is even valid. The question which may be more appropriate is "When, if at all, will the revolution begin?"

The second point I wish to make is that if we assume for the moment that the revolution has in fact begun, then its prognosis for success is very slim. The reason I would argue this is that the very well intentioned people who are promoting the revolution are seriously underestimating the difficulties involved. These difficulties are the result of many factors among which may be numbered the necessity for teachers to reconceptualize their view of education, of children and how they learn, and of the nature of Logo. In my view, if the revolution is to occur, then teachers have to know a lot more Logo than most presently do, and in addition, they have to have a view of education and children which is antithetical to *schooling* as we know it in North America.

Point three arises from an examination of the preparation which most teachers have received prior to introducing Logo into their classrooms. Not even an introductory, semester long course on Logo will necessarily prepare a teacher to introduce Logo in ways which would foster the revolution. Knowing Logo as a language is a necessary but not a sufficient condition to implementing it in one's classroom. Much more is required, and it is that 'much more' wbich by-and-large has not been developed in any teacher inservice on Logo that I'm aware of as a result of reviewing the reports of people performing such activities. Most reports one reads state that one goal was to consider, examine, reflect on the Logo philosophy, but tbat once tbe inservice is underway, most of the time and energy was devoted to learning Logo, the language. It should not be surprising tben tbat the revolution bas not begun or at best seems bogged down.

The fourth point is a variation of the old 'saw' that if one doesn't know bistory then one is doomed to repeat it. Those of us who have lived through any major curricular revision or movement—and I specifically have tbe New Math movement in mind here—realize the significant similarities between what is happening to tbe introduction of Logo and the results of the New Maths revolution. Even the inservice formats are similar in that pyramid style organizational models are used to "train" Logo teachers who tben "train" more Logo teachers and so on. This is precisely the model which Bob Davis used in tbe Madison Project, for example. It is fairly well documented that what happens in such situations is that the superficialities are what gets passed along while the "guts" of the proposal get lost. Hence, it is not surprising that we now see on the market Logo workbooks which provide systematic instruction on bow to draw a square, a procedure wbich every child is supposed to go through, step-by-step. Such an approach to Logo is directly analogous to a teacher using Cuisenaire rods and saying that the white rod is one, the red rod is two, and so on.

The fifth, and final, point I wish to make is rather heretical. It is this: Logo the computer language is not the important aspect of the Logo revolution! The philosophy and psychology behind the Logo language is the crucial thing. The language itself is a vehicle for the propagation of an approach to education which honours the integrity, power and creativity of children. That is why it is so important Logo advocates to constantly seek to clarify, elaborate and explicate those ideas and conceptions about children which guided Papert and his colleagues to develop the Logo language as a means for children to express themselves. If the revolution falters, either in actually getting started or in maintaining whatever momentum it has acquired, it will he because the advocates of Logo have not been able to express their philosophy in ways which teachers can understand. Moreover, an important component of that understanding must be that teachers see it as being in their self interest to create and function in a Logo environment. It is only in a teacher's self interest to do so if functioning within a Logo environment is energizing to that teacher, enabling for the children with whom the teacber is working, and still allows the teacher to meet community demands for a sound education.

The reason that Cuisenaire rods and the New Math did not make a significant impact on American education was not because they were inappropriate or wrong headed innovations. The reason was because the advocates (and I don't use that word in any perjorative sense) were unable to demonstrate and convince teachers that the adoption of those innovations was in their self interest. I fear that advocates of Logo will suffer the same fate if they don't address themselves to the task of showing how the creation of a Logo environment and the use of computers and the Logo language will serve teachers very well in achieving the goals they have set for the children in their charge.

For the revolution to occur, a teacher must establish a culture in the classroom which, as I said before, honours the integrity, power and creativity of children. That culture must honour the learning styles of the children and the teaching style of the teacher. Perhaps most importantly, teachers have to subordinate their teaching to the learning of the children. Without such a surrounding culture, the Logo revolution will not begin, or if begun its momentum will not be maintained, and it along with the Cuisenaire rods will be relegated to the closet of discarded educational innovations only to be brought out as "interesting toys" to be used after the important things have been dealt with in the classroom. If the revolution is be successful, then it is the creation of the culture Logo advocates must focus on, and not the language aspects of Logo.

Each of these points will be fully developed and defended in the presentation.

Thousands of Mules

This session will consist of excerpts from the electronic correspondence of the presenters — a correspondence spanning several months, several thousand miles, several networks, and several points of view.

It all started in Paradise Valley at the World Logo Conference last October. A typo grew into a joke and then into a cliche, and finally ended up as the focal point of an electronic discussion about Logo communities, networking, and the metamorphosis of bugs into butterflies.

Topics which will be addressed during this discussion include:

(a) The importance of humour and ritual in the evolution of organisms, organizations, and communities.

(b) The need for a strong autopoietic Logo community

(c) The role of the audience in the transformation of bugs into powerful ideas

(d) Teaching styles in the Logo classroom — stabilizing and destabilizing tbe learning environment

(e) Appropriate use of technology — beating back the tide of technocentricity

(f) Microworlds as an art form — the aesthetics of programming in Logo

(g) Proof by contra-dancing — a treatise on the need for more levity at Logo conferences

Presenters:

Brian Silverman
Logo Computers Inc
9960 Cote de Liesse
Lachine, Quebec
Canada

Gerri Sinclair
Faculty of Education
Simon Fraser University
Burnaby,B.C. V5A 1S6
Canada

SUSAN WOLFF
1608 STOWE RD
RESTON VA 22094

Learning Through Logo

Susan Wolff
Fairfax County Public Schools

The question proposed of which Logo to teach, graphics ot text oriented, is an interesting one. Two years ago I needed to address this question. I found that my decision was largely influenced by the amount of on-line time my students were able to have, a factor that is similar in most schools.

When the typical classroom teacher plans his/her Logo time, and needs to condense a Logo lesson and/or on-line time to the one or so hour a week allotted, decisions truly do need to be made about what will be given top priority.

When, in 1984, I initiated my Learning Through Logo program at my school it was mainly because I decided I didn't want to choose. I wanted it all. I wanted to try using the many facets of Logo with a 5th grade class and see how far these children could go. So, I begged, borrowed, and bought enough computers to support a Learning Through Logo program independent of my school's computers.

Last year, at Logo '85, I showed a videotape of the software my class had written. They used Logo to write and illustrate programs in social studies, science, math, research on subjects they chose, reading, and much more. They used the text capabilities of Logo as a word processor, the deal being that when they learned a skill or some interesting facts in a particular subject area, they could use Logo to write software to teach the lesson to someone else. We had wonderful software on a variety of subjects. If they learned about an explorer they could write a program about that explorer. When they learned about nouns, verbs, and adjectives, they wrote madlibs using variables.

There were three advantages to my Learning Through Logo program. First, through the use of the text capabilities of Logo, the children's written skills improved dramatically. Logo became their word processor. There is much more at stake for a child writing a report that will be saved on a disk and shared with others, then when it is written down and just handed in to the teacher. In the latter case, even if the report was read aloud, no one but the teacher would probably read it. But, when your information is up there on the monitor for everyone to see, you really do want it to look nice. And when it is so easy to edit and change your work it becomes very non-threatening. The children did a wonderful job of proofreading their own and each other's work. They worried about spelling, complete and run-on sentences, and how things look. They took much more pride in their work and really cared about the finished product.

The second advantage of my Learning Through Logo program was what the children could do with their Logo graphics. They thrived in the world of the turtle because the wonderful pictures and animation the children created could be placed in the context of a program. When they were writing on-line "choose your own ending" adventure stories, they made wonderful illustrations for their stories. The children had all the advantages of discovering what the turtle could do, and all the mathematical thinking involved in playing turtle. They loved it, and when they were done, their finished product became part of a program they could share with others.

The third main advantage of using all the parts of Logo, is really the result of the first two.

Using Logo to integrate social studies, science, a writing program, and including Logo graphics, sold my program. How many times do we hear teachers say they don't have time for Logo as "one more thing to teach"? I was able to show how Logo could be used extensively to help with areas we are already teaching, in a way people could really see. In my classroom it was not an add-on, but the way the children learned and showed proof of their learning. There were two groups sold by this program. Those in charge of curriculum for the county were convinced, because of several Logo projects in the county that Logo could easily be integrated into our Program of Studies, and indeed even came up with a grade level scope and sequence for Logo. The other group was teachers. Those teachers that came to observe my class left feeling really good about using Logo in the classroom. The county even let me teach a course on integrating Logo into the curriculum to about 20 Logo-using teachers last semester. Those teachers have been using Logo extensively in their classrooms. This is no small matter, because spreading good feelings about Logo is important. It's what's going to put more computers in the schools, and get them used by teachers. The benefits to the children in the schools are important.

The results of last year's program were so great, both in academic and social areas, that this year the Learning Through Logo program at my school was extended to include the entire 5th grade. I have also been given the opportunity to act as a computer resource teacher and help other teachers use computers in their classrooms.

At Logo '86 I would enjoy speaking about both the first and second year of this program that took advantage of both Logos, as well as show some sample software written by the Learning Through Logo students.

Can We Authentically Integrate LOGO into the Elementary School Curriculum?

Dr Chris Templar
Johnson Bible College
Knoxville, TN 37998

This presentation seeks to address some of the problems associated with integrating LOGO into the traditional elementary school curriculum. With the increasing emphasis on competency based programs, assessment outcomes, and basic skills many elementary school teachers are feeling pressured and threatened. The idea of finding time for a discovery oriented computer language appears to many to be just one more pressure for which they are neither equipped nor competent.

One option which is becoming popular is the idea of using LOGO with traditional subject areas. Many of the ideas which are suggested appear to be denying the spirit of LOGO and almost fettering it within the bonds of traditional curriculum. This paper will consider whether this is inevitable. Examples will be presented from K-6 curriculum guides which were developed for Knox County schools.

This curriculum sought to address two major problems. These were: First, how do we allow for a large number of students to be able to explore within LOGO environment when only about 25% of their teachers have had exposure to LOGO? Second, how do we add LOGO to an already full day.

The State of Tennessee Department of Education emphasises test scores and issues "report cards" for school districts. This has significantly increased the pressure being felt by both children and teachers. In the light of this a multi-faceted approach was made to the curriculum development.

This presentation will focus on the issues and problems encountered and attempts that have been tried as well as some solutions that have been found to some of these problems. It will not be a description of curriculum guides. It is the hope of the presenter that the results of some of the research that has been conducted into ways to effectively integrate LOGO into the curriculum whilst maintaining the authentic spirit of LOGO will be of help to other teachers and curriculum developers.

A MUSIC CLASSROOM IN THE LOGO SPIRIT

Rena Upitis
Faculty of Education
Queen's University
Kingston, Ontario
Canada K7L 3N6

This session will describe the music environment at a special inner city school in Jamaica Plain, Massachusetts. The school is the major site for an ongoing large-scale research project conducted by the Learning and Epistemology Group of the Arts and Media Technology Laboratory at MIT. The project (called Project Headlight) was conceived with a view to building a "School of the Future", where a large number of computers (approximately 100 computers for 200 children) would be used in an open educational setting for child-centered learning.

I am one of a team of teachers and researchers taking part in Project Headlight. My role has been to create a music environment in which children are able to make music to communicate messages, stories, ideas and feelings. Music is unique in the Project Headlight school. Not only are computers an important feature in the music classroom, but music itself is an integral component of the school community.

While it is true that computers play a significant role in the Project Headlight music program, it is also true that the computer presence does not solely account for the success of the music program. If I were to identify the one most important aspect of the music program, it would not be that children use computers, but that they create their own music. That is, children are expected to improvise, compose, and perform. The computer gives children one strong tool for achieving these goals. Further, by using the computer as one of a number of materials for making and understanding music, the computer has come to be viewed not as something different or unique, but rather, as a tool which is good for some purposes and not for others.

As more sophisticated and powerful music software and hardware becomes available for elementary school children, it is imperative that we learn how to create living musical environments for such computer tools to have full impact. There are two reasons for this. First, most children spend little time playing with musical ideas and creating original music. (Note that this is in sharp contrast to a child who first comes in contact with a graphics language, since virtually all children have tried their hand at drawing or painting; likewise for word processing programs — most children have experience at writing stories from a very early age). Thus, teachers will first need to provide a "musical playground" where children can improvise and compose without a computer. Then, when children approach composition in a procedural fashion at a computer, they will already have some ideas about how musical segments can be transformed and manipulated based on their earlier explorations. The second reason for learning how to create a dynamic musical environment is that the computer alone cannot provide a full musical experience even if children have first learned about music away from the computer. The computer cannot replace what children learn through their own bodies, nor can it provide the kind of learning which is involved in playing a classical instrument. Rather, computer-based learning along with other open-ended music settings serve to enhance one another, and together contribute towards building a living musical environment. This type of musical environment is, in fact, a "Logo environment" in the fullest sense: a place where a child can direct his or her own learning in a meaningful way that leads to deep personal insights into the very heart of the subject of study.

The presentation will focus on ways of exploring music that do not involve using a computer, but that nevertheless contribute to building a "Logo environment" in which computers play a vital role. It will be shown how many of the described activities were developed from the materials and problems provided by the students themselves. For instance, the use of children's invented notations as a link to standard music notation will be discussed. In addition, movement techniques for encouraging improvisation and exploring musical texture and form will be described. Improvisation techniques for simple instruments such as bells, and more complex keyboard instruments will be featured. The presentation will also feature a description of settings for the performance of children's compositions as well as the works of others. Finally, ways of relating music to mathematics, language arts, and the social sciences will be addressed.

Logo and Mathematics at Agnes Irwin
One try at putting the theory into practice

Richard Binswanger
Agnes Irwin School
P.O. Box 407
Rosemont, PA 19010

How can Logo be implemented in a traditional high school without sacrificing Logo ideology? Can it be effectively integrated into an educational environment with an established set of competing philosophies? Can teachers and students used to education by lecture embrace the more unstructured style and exploratory approach which Logo offers? Such questions barely occurred to me, when I began my work with this language four years ago. Logo philosphy so appealed to me that I did not understand how anybody could not get caught up in it once they understood it. In some ways, I was naive. Despite my efforts to inform and arouse enthusiasm , many of my colleagues remain relatively unimpressed with Logo. And yet, there have been major strides and I am more convinced than ever that Logo does have a place in my high school. But I now believe that the language must adapt to its clientele as well as vice versa.

Two years ago I began an effort to integrate Logo into the standard math curriculum. I already had instituted Logo as the language in the introductory programming course and found it to be an excellent choice in teaching novice programmers. But I came to question the desirabilty of teaching programming per se to all students. Still, I was convinced that Logo was an ideal tool in getting students to think in intriguing ways and play with mathematical ideas. Last year I reported our progress at Logo 85 and this year I would like the opportunity of presenting an update. A most significant shift of perspective has taken place this year. Our emphasis has moved away from the language itself and onto the people who use it.

I work at the Agnes Irwin school, a small K-12 girls´ school in the suburbs of Philadelphia. We are blessed with extraordinary computer facilities with a computer to student ratio of about 6 to 1. Even so we struggle to find appropriate and intriguing ways to use our machines. Integrating Logo into the mathematics curriculum has been my department´s highest priority. The math department I think is somewhat typical of a college preparatory school facing the pressures of making sure that the students cover enough material before they graduate. The department´s chairman felt hesitant in devoting time to something so ´experimental´ as using Logo in a high school mathematics classroom because it seemed risky to him. He gave us a chance though, after we argued that our first priority was to promote mathematical thinking in our students using the concepts from the standard math curriculum. So I set out with an enthusiastic member of the math department to institute a desirable program. We began with the following criteria:

1) Logo should become a standard part of the mathematics curriculum K-12 and should reinforce the material, we presently taught.
2) We were interested in using Logo in ways that would allow our students to explore mathematics as opposed to witnessing demonstrations.
3) We initially should work to minimize the amount of programming that students would have to do. We worried that their difficulty handling pure programming concepts and their akwardness with Logo syntax might get in the way of their learning the mathematics. To this end, we tried to create procedures and functions ahead of time as tools for student explorations, in some sense trying to build a mathematics microworld. As students become more accustomed to the language, more programming concepts could be presented.
4) We targeted the geometry courses as an ideal place to emphasize our Logo program, as we immediately saw many varied uses of the language in that area.

To institute the above, I began team teaching sessions with different math teachers in the computer lab. Students worked on the computers either individually or in pairs. We have had good success with many classes, yet interestingly enough in practice all four premises in some ways worked against us.

1) We didn't have the personnel to institute Logo in the entire mathematics curriculum, but as a stated goal we perceived ourselves as failing due to the slowness under which change took place. Even in the geometry curriculum, we might have stretched ourselves too thin.
2) In trying to promote discovery and exploration, we often frustrated our students. They complained that they had no basis from which to start working and had difficulty connecting their computer work to their work in their regular math room. We expected and even hoped for a certain frustration level, but they were giving up too often. The shift toward doing and away from listening was new to them and many simply dealt with it poorly. This lead us back to introducing more structure in an effort to gradually ease them in to this experimenting approach. But that too had drawbacks.
3) In trying to simplify the protocol to allow the students not to get bogged down in the language itself, we sometimes provided procedures and functions that may have been too powerful and questions that may have been too leading, reducing the work to a cookbook routine. Trying to strike a balance between giving too little or too much help and information is the key dilemna that faces any teacher that values the Logo philosophy.
4) In targeting geometry, we asked and received a sixth class per week from the administration. The head of the department believed an extra class was needed, so as not to jeopardize the existing geometry curriculum. This I believe was our greatest, but most subtle blunder. In theory, students had six geometry classes a week, They did not perceive it that way, but rather as five geometry classes and one computer class. The computer class was considered something extra, meeting in the afternoons, and seemed somehow unrelated to their morning geometry class. It was an imposition. We ourselves often reinforced this idea, when we tried to get everything done in the afternoon class instead of teaching concepts and allowing the possibility of doing several sessions in a row. We even gave them afternoon study halls on days when we decided not to take them into the computer lab.

I consider this last point to be absolutely crucial. I have taught some extremely successful classes using Logo in courses ranging from pre-algebra to calculus and they were always during the math class' regularly scheduled class period. We have even taught identical lessons to different geometry classes, one during the afternoon lab session and one during the regularly scheduled class and have enjoyed much greater success with the latter. The students expected to do mathematics and they were eager and appreciative for something different. We had them in a decent frame of mind.

The approach we have developed is a cross between completely free exploration and more standard teaching (learning) practices. We would begin with specific objectives the students could attack. Sometimes we found it helped, when a first project was worked on by the class together. Ideas were exchanged and confidences were built. They were comforted by the fact that they were not alone in their uncertainty. Directed questions were asked to draw students out. First reactions to this probing were typically negative. "Why ask me, You know I don't know.", to which one of us would respond with something like, "That is exactly why I asked. I want you to think about this. You certainly wouldn't have to think if you already knew the answer." We certainly encouraged our students at every opportunity and even at times pressured them to continue trying, maintaining that anything short of that was unacceptable. We however did not dictate their approach nor preclude them from walking away from a problem for a while. Most importantly, we sensed our job as teachers was to ask questions and give some direction, but not supply answers. We believe our students needed that structure to get to a point where they felt more comfortable with the process. As fast as we could, we backed off. We began to ask more open questions and worked towards more generalizations, in essence letting their minds and abilities take over. We tried to finish class with the question, "Where could you go from here?", encouraging them to generate questions and avenues of exploration. They discovered that one problem's solution is often a more fundamental or general problem's creation.

In my presentation, I would expect to simulate and discuss our approach with a specific example, probably an introduction to Cartesian graphing. I would show the specific procedures we gave to them as tools. I would work to recreate the environment and give the progression of questions that we used, as well as give anecdotes of the student's reactions.

We are excited by what we have seen and done. Some of the most vibrant math classes that any of us have ever taught have been in the Logo lab. Our students are playing mathematician, but often the effect is temporary and limited. Students who finished a lesson motivated, probing, and stretching themselves often revert to simply waiting for answers when they reenter the lab with a new project. A major problem that we face is that the philosophy of the lah is not being utilized or reinforced elsewhere. The teachers find it difficult and often inappropriate to carry over the style of the lab back into their classrooms. Change can be scary, especially one that attempts to reduce their established structure and thus threatens to expose their weaknesses. Throw in something new like the computer and the job is doubly difficult. They need to learn to view the computer as simply a tool, but more fundamentally I'd like to see more flexibility in their styles and a shift in emphasis away from teaching and onto learning. So where do we go from here? The irony is intriguing. At the core of Logo philosphy is the idea that although it can be learned, it can't be taught in a traditional sense. So we are now caught in the problem of MetaLogo, that is how to facilitate teachers into discovering and applying the philosophy of Logo. Our approach has been Logolike. We have set up weekly meetings of the math and computer departments to discuss the integration of Logo into mathematics. We use these discussions as much as possible to get at the fundamental reasons and ways the teaching of mathematics should take place both with and without Logo. And by team teaching, our styles I know rub off on each other. I know my perspective is changing even as I work to change their's, but I am thankful for that. Working with the other teachers has been my most exciting and frustrating part of this attempt to date. Things move slowly, but I think progress is being made, in a large part because I work with teachers who care. To inculcate the Logo philosophy in them involves again a fluid structure. I would like the chance to explain this structure as well at the conference. I see my personal interpretation of the Logo ideal going through a metamorphosis. It is becoming more pragmatic. I hope we do not subvert the original intention. I don't think we are, because our students and teachers are focusing more of their energies on the process of thinking.

USING LOGO IN SENIOR MATHEMATICS

SEBASTIAN REISCH
CLARKE HIGH SCHOOL
NORTHUMBERLAND AND NEWCASTLE BOARD OF EDUCATION
NEWCASTLE, ONTARIO CANADA L0A 1B0

Grade eleven and twelve computer science students were introduced to LOGO in order to broaden their programming experiences. Several of the programming ventures turned into potentially useful utilities, which after further work, evolved into microworlds or toolkits with which students may pursue mathematical concepts at the senior level.

As a result of these experiences, these students seemed to gain a much deeper understanding of the mathematics involved than they normally would have in a more traditional mathematics class.

This increased understanding having been noted, the microworlds were introduced into some of the math classes to be used as an alternate teaching strategy. Most of the students in these classes were already familiar with LOGO. Those who were not, received several familiarization sessions. The students were to fulfill curriculum specific tasks. They were usually successful at completing the task. In the process, however, they often came across unexpected results which they had to pursue in order to satisfy the requirements of the assignment. The toolkits usually facilitated this pursuit and in many cases the student actually felt prompted to follow new lines of exploration before carrying on with the completion of the task.

It was observed that these spontaneous lines of exploration contributed greatly in giving the student a deeper understanding of the mathematics s/he was learning.

The microworlds which have been used experimentally relate to the following curriculum topics:

a. Algebra of functions
b. Graphs of functions
c. Euclidian geometry
d. Transformations of geometric figures
e. Transformations of functions
f. Vector geometry and vector algebra
g. Matrix algebra
h. Probability

From these initial experiences, it was recognized that such toolkits may be an important ingredient in an unusual but potentially fruitful learning strategy.

Since then, the most promising of the kits have been or are being packaged in the hope that they may become useful on a more regular basis.

The criteria for packaging should include:

a. a well designed structure so that each toolkit is simple to use and may be easily modified,
b. command sets that are consistent throughout the series of toolkits,
c. a facility to extend the command set of each kit,
d. a facility to store and retrieve language extensions and students' work for each kit,
e. appropriate printer utilities for each kit,
f. thorough documentation on how to use each kit,
g. a starter list of pedagogy and curriculum targets.

AN OVERVIEW OF SEVERAL KITS.

The experimental kits which have been packaged are TRANSFORMATIONS OF GEOMETRIC SHAPES, GEOMETRIC CONSTRUCTIONS, FUNCTIONS AND GRAPHS and TRANSFORMATIONS OF FUNCTIONS.

Each toolkit is an open microworld, i.e., it consists of a set of pre-defined procedures. The toolkit is read into logo's workspace thus extending the vocabulary of the language. The user then has access to all of LOGO's repertoire of commands, as well as those of the toolkit; in fact, the user has the option to further extend the language by using the appropriate command built into the toolkit.

TRANSFORMATIONS OF GEOMETRIC FIGURES is a microworld which provides the students with the means of pursuing the ideas and concepts of transformation geometry without having to go through the time consuming details of the transformation constructions. With this toolkit, the student performs a sequence of transformations on a geometric figure, observes the orientation of the result and formulates conclusions. At all times the student decides the direction of the line of exploration.

There are three basic types of transformations that may be done within this toolkit: transformations in the plane, rotations about the centre of the system and flips (reflections) about lines which pass through the centre (origin) of the system. Further transformations, which are compositions of the basic ones, may be built by the student.

GEOMETRIC CONSTRUCTIONS is also an open microworld. It provides the students with the means of pursuing the ideas, concepts and structures of Euclidian geometry. The student constructs geometric situations, hypothesizes about these situations, verifies the hypotheses in similar situations and generalizes the conclusions by defining new construction commands.

FUNCTIONS AND GRAPHS was built to provide an environment in which the student can play with and explore the behavior of functions and their graphs. The drudgery of number crunching is being done by the toolkit; discovering the functions' characteristics and making decisions about the course of action is done by the student.

The student makes decisions about:

a. The domain of the function and its exceptions, such as division by zero, complex number situations, undefined function values, discontinuities, etc.
b. The size of the increment and scales values to be used in order to discover all the interesting contrasts of the curve of the function.
c. The boundaries of the interval over which the function is to be graphed so as not to get the 'wrapping' effect on the screen.
d. The appropriate use of parentheses when defining a function in order to ensure proper order of operations.

TRANSFORMATIONS OF FUNCTIONS is a microworld which applies geometric transformations to the graphs of functions. The student extends her/his knowledge of functions and their graphs by observing the results of transforming the functions. The basic transformations available in the toolkit are rotation about the origin and reflections about lines passing through the origin. The matrices which represent the transformations mathematically are represented in the toolkit by lists of lists, where the sublists represent the rows of the matrices. The student has the option to create further transformation matrices.

Introduction to Object-Oriented Programming in Logo

Laurence J. Davidson ***and***
Philip G. Lewis

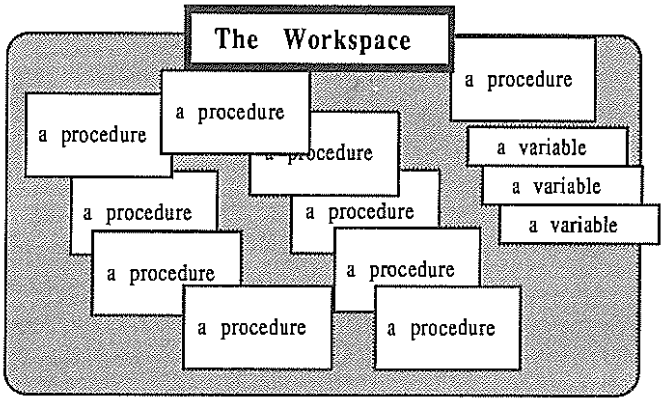
Computers for a New Education

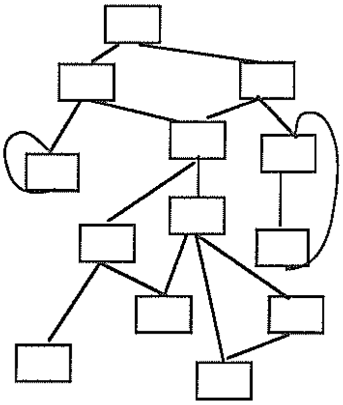
Several participants in last year's **Logo 85** Conference raised interesting questions concerning future developments and directions for Logo. Here we are now, a year later, and the future is on our doorstep. At least one of the proposed future developments is ready to be learned and used: **object-oriented programming**. Its vehicle is Object Logo.

Our talk will be an *introduction* to Object Logo, aimed specifically at you if you are familiar with one or more of the traditional versions of Logo but have no experience with object-oriented languages. We will use concrete examples to show you what Object Logo is all about; we will not be discussing the philosophy and theory of object-oriented programming *per se*, but some consideration of philosophical issues will grow out of the context of the examples.

In this paper we don't have space for consideration of a lengthy example, so some general words will have to suffice. A more detailed paper is available from the authors upon request.

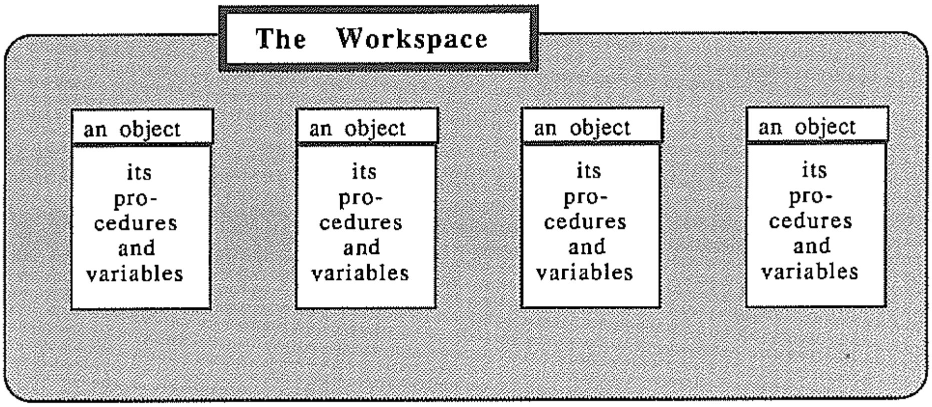
Object Logo is an extension of Apple Logo—and yet much more than an extension. It is an extension in the sense that a program written in Apple Logo will run in Object Logo with little or no modification. But it is much more than an extension in the sense that the object-oriented approach changes the whole way we organize and think about programs. Consider what the workspace looks like in a large traditional Logo program. It's a clutter of perhaps dozens of procedure definitions and global variables, all arranged according to no discernible principle, and even a procedure-calling tree doesn't help much:





In traditional versions of Logo—as pictured in this illustration—your world consists primarily of **verbs**. Procedures, after all, are instructions ***how to do*** things: `TO` this, `TO` that.... But nothing says that one set of verbs conceptually belongs together in one place and that another set belongs together in another place.

The object-oriented approach organizes your workspace and your thinking so that procedures and variables that belong together stay together. It adds **nouns** to your world. You specify not only ***how*** to do things, but also ***who*** knows how to do what. The "who" in each case is what we mean by an **object**. Here is an illustration of the object-oriented organization of a workspace:

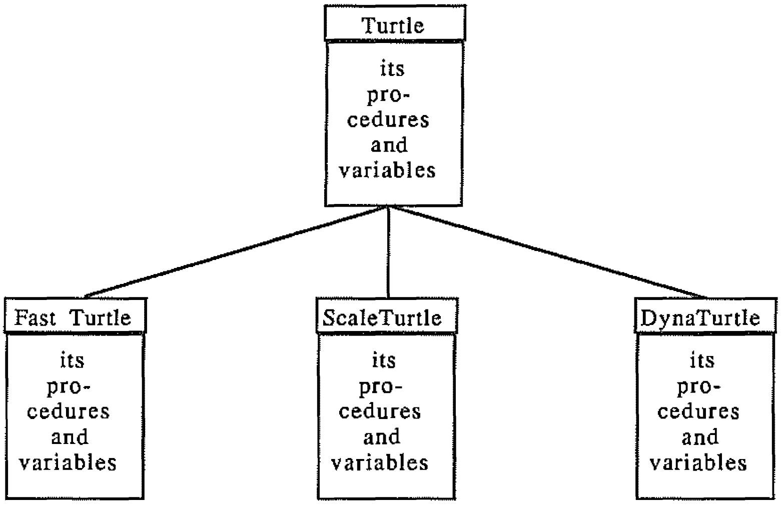

The Workspace
an object
its pro- cedures and variables
an object
its pro- cedures and variables
an object
its pro- cedures and variables
an object
its pro- cedures and variables

Actually, the object-oriented organization is even more structured than is pictured above. For not only are procedures and variables organized into objects, but objects themselves have an organization. When you program in Object Logo you develop a **hierarchy** of objects. If you have three types of turtles—a fast turtle, a scale-drawing turtle, and a dynaturtle, let's say—you don't need to repeat in each object all the procedures that they have in common. You merely say that each is a kind of turtle, with instructions like

```
Make "DynaTurtle KindOf  :Turtle
Make "ScaleTurtle KindOf :Turtle etc.
```

Now each kind of turtle automatically inherits all the knowledge of its conceptual parent, the generic `Turtle`. Only the differences need to be specified. This characteristic of most object-oriented systems, known as **inheritance**, does far more than save keystrokes. It provides a conceptual organization that allows us to structure our thoughts in appropriate ways.

For example, if we want to add a new procedure that all kinds of turtles should be able to do—e.g., `Reverse`—we add it to the `Turtle` object and the other kinds automatically inherit the behavior. On the other hand, if we want to add a new procedure that only dynaturtles should be able to do—e.g., `Accelerate`—we add it to the DynaTurtle object and the other kinds know nothing about it. You might picture the hierarchy of turtle objects as follows:


Turtle
its procedures and variables
Fast Turtle
its procedures and variables
ScaleTurtle
its procedures and variables
DynaTurtle
its procedures and variables

This diagram is an attempt to capture an important generalization: a fast turtle is exactly like a generic turtle, knowing the same procedures and variables, except for certain explicitly mentioned differences. Similarly for a scale-drawing turtle and a dynaturtle. Thus these objects are alike in some ways and different in others.

We mention turtles only because they're a familiar kind of object to all Logo programmers. But objects need not be so familiar nor so concrete. Windows and files are objects, knowing such behaviors as `Close` and `Open`; fractions might be objects, with their own special knowledge of `Sum` and `Product`; even an abstraction like a calendar might be an object, containing procedures for updating itself and printing your appointments. Whenever you have a large program that can be structured so that certain procedures and variables belong in one place and other ones belong in another place, you have a good candidate for object-oriented programming.

AN ELECTRONIC NETWORK FOR SUPPORT OF LOGO APPLICATIONS FOR SPECIAL POPULATIONS

Glen Bull and Paula Cochran
Curry School of Education
University of Virginia

David Cartmell
IBM Corporation

Cheryl Wissick
Albemarle County Public Schools

The problem faced almost immediately by a clinician or teacher planning to use a computer with a handicapped child is finding an appropriate activity. Each child is different, and menu-driven educational software is often inflexible. We wanted to use Logo to address this problem.

Despite some cleverly designed projects developed by and for teachers, and a few Logo converts who have really blossomed and made Logo a key part of their teaching repertoire, the results were initially not as promising as we had hoped. It developed that teachers and clinicians did not just "know" what to do with the Logo skills they had gained.

Our special educators and clinicians in communication disorders floundered -- they tried to teach what they knew. What they knew was Logo commands. They were often surprised to find that their language-delayed and/or mentally retarded children were NOT empowered by lots of typing with an occasional response from a small triangle of light on the screen.

Solutions?

After several semesters we began to shift our emphasis. The teachers we had taught were beginning to call and ask us to write specific procedures for a particular educational purpose. Very often they knew what they _wanted_ the computer to do, and their Logo experience had taught them what the computer _could_ do. They needed a little extra programming expertise. We began to provide key Logo procedures, which we began to consider "tools for special education".

At present we are in mid-thought. We have reached the following tentative conclusions:

1. Teachers and clinicians often do not learn enough Logo in a traditional one-semester course to devise their own classroom applications, much less in one-day in-service orientations. After they return to the classroom, they frequently spend their time turtling with the children. After a trial period of experimentation, many lay Logo aside altogether.

2. The problem, as we see it, lies in making these teachers start from scratch.

3. From our perspective, the most powerful component of Logo is its ready extensibility and open-endedness. That is what initially captivated us, and what sustains our interest.

4. Providing a teacher with Logo tools (key Logo procedures) uses Logo's extensibility to good advantage. The archetypical example is PICK. This procedure abounds in the Logo literature. Most teachers will never spontaneously develop or discover this procedure. We think that it is not particularly important that they understand how it works internally, as long as they know how to use it in other procedures.

   With PICK and the proper template (model), a teacher who has never used Logo before can create her own computer-generated poetry in half-an-hour. Further, with a little more assistance, she can transform the model into other poetry forms she can conceive (as well as a number of other language forms such as mad libs).

5. People who have not used Logo (or who have had only minimal exposure) have a difficult time comprehending the empowering nature of an open-ended tool. They suggest, for example, that the illustration of poetry generation that we cited above could be turned into a menu-driven application, with word banks entered by the teacher in response to prompts.

   The situation is reminiscent of early reactions to word processing. In the mid-seventies it was difficult to describe the power of word processing to people who had never used it. It was only after people experienced word processing that they understood (and immediately became excited).

Current Logo Use with Special Populations

Not everyone who works with the handicapped uses computers (although the number is increasing daily). Some evidence suggests that about ninety percent of those who use computers in special education do not use Logo. The specific numbers are not important; the point is that the percentage who do use Logo is small.

The majority of the ten percent (or so) who use Logo tend to use it for turtling. The remaining few use Logo in creative and innovative ways with their students and clients. We have seen some clinicians of this type working with the communicatively handicapped.

The pyramid of Logo use with special populations looks like this:

90 % do not use Logo
10 percent use it primarily for turtling
1 % use Logo as an effective tool in a variety of applications

The question is: "How do we invert the pyramid?". The problem facing us is the same one facing the general Logo community. How do we teach and transmit truly useful educational applications?

A Network for Special Populations

We observed what we considered to be two key factors. The first was that most of the good Logo applications for special populations in our school district were generated by the needs of the teachers themselves. Often a teacher had a clear concept of the application and only needed a key procedure or two to implement it.

The second observation was that once an application developed, it was often transmitted from teacher to teacher. At each step along the way the key procedures or tools were modified, adapted, and enhanced to meet the needs of the individual user.

We wondered how these applications could be transmitted beyond the immediate geographic area. One prior experiment involved compilation of applications in a manual accompanied by a disk of procedures (Bull and Cochran, 1985). However, the mechanics of addressing and mailing a manual every few days quickly become cumbersome. Further, the manual effectively froze applications at a particular moment in time. In theory the compilation of applications could be updated and revised, but in practice there never seemed to be a convenient time to do this. Further, provision of support to other geographic areas through long-distance phone calls became expensive and time consuming.

We considered whether an electronic network might make it possible to transmit Logo applications more freely. In a network of this type, it is as easy to query ten individuals as it is to ask one person. Respondents can answer at their own convenience, rather than when the caller telephones. Further, the Logo procedures themselves could be transmitted electronically. Thus, it might be possible to transmit a defective procedure to a colleague as part of the debugging process.

To initiate this network, a month-long institute was scheduled to bring together special educators and clinicians from geographically disparate locations. The intent was to establish a common core of applications, which would then be expanded and enhanced through interactions at a distance. The preliminary results of this institute and subsequent interactions through the network will be discussed in the session.

Logo with Special Education Students

As part of a Master's of Science program in Computers in Education at C. W. Post, Long Island University, this writer conducted an eight week research project using Logo with learning disabled and emotionally handicapped children.

The research began five weeks after the start of the 1985-1986 academic year. Specific areas studied included knowledge of geometric properties, emotional and social development, and awareness of, and changes in, problem solving skills.

The research conducted involved a survey (pretest and posttest), group and individual instruction, teacher observation, student-teacher interviews, and students recording their reactions to working with Logo.

Classroom techniques employed by the teacher included: class lecture and discussion, individual instruction, grouping students together to work on projects, physically demonstrating an idea by having a student "walk" through a procedure, and having students draw their ideas on paper, write the commands that matched their ideas, input the commands on the computer, and then attempt to debug them if necessary.

The students attend a B.O.C.E.S. (Board of Cooperative Educational Services) program in a regular junior high school in Suffolk County, Long Island, New York. Nine students, ages twelve through fourteen, were

Shelly Bernstein
Long Island University
97 Fairview Ave.
Deer Park, NY 11729

used as subjects. Two were female, seven were male. I.Q. scores (from the WISC-R) ranged from 69-96. The classroom contained one Apple IIE computer, and the Terrapin version of Logo was used.

Based on a review of the literature, three hypotheses were formulated: Hypothesis 1: Instruction in Logo will increase a student's knowledge of geometric properties; Hypothesis 2: Instruction in Logo will have a positive eff-ect on a student's emotional and social development; Hypothesis 3: Instruction in Logo will increase the number of ways a student might try to solve a mathematical or spatially oriented problem.

Results for the project were reported as case studies on each of the nine students. The first two hypotheses were confirmed by the brief study; the third one was not confirmed. Background information on the teacher (as she was an integral part of the study), as well as some unusual circumstances which arose during the project, are discussed in the "methods" section of the paper.

Logo Programming Style:
A Contradiction in terms?

Sharon Burrowes
Wooster City Schools
Wooster, Ohio

Most of what is written about Logo describes the open environment in which students can experiment and explore. Much is made of the creative expressions and exciting discoveries. However, there is another side to this "rosey" picture. What of the student with a particular goal in mind who finds a bug s/he can't fix? The excitement can, for some, quickly turn to frustration or defeat. Not infrequently these elusive bugs can be traced to poor programming style or design.

But wait, if the Logo environment is truly exploratory and experimental, then is talking about programming style a contradiction in terms? If we impose "rules" for writing Logo programs, are we infringing on the creativity of students?

As a secondary computer science teacher, I have spent many class periods discussing the importance of careful program design and good programming style in both Basic and Pascal. As Logo moved first into our elementary schools and then later into the high school, I naturally found myself developing a Logo programming style that reflected all that I had been teaching about structured programming, and then passing these ideas on to my students.

It wasn't until several colleagues said of my Logo style: "Logo isn't done that way" that I realized that my style was in any way unusual. Indeed, when I examined a number of articles and books, I found that almost all elementary Logo is written somewhat differently from the way I had come to write Logo programs. My own style had grown out several years of computer science courses all preaching structured programming and embodying their own Pascal-like pseudocode. Some informal research led me to the conclusion that much of the existing Logo programming style has grown out of necessity. Programmers working with early microcomputer versions of Logo were restricted by the limitations of these early versions of Logo. Thus, today's "accepted" style reflects restrictions that do not exist in some of the new versions of Logo.

This discovery caused me to raise a number of questions about my own Logo programming style as well as my teaching of Logo programming: Is it appropriate to impose stylistic restrictions on students when teaching a language such as Logo? Is it possible to teach about programming style and design without imparing the creativity of students? At what age should stylistic matters be addressed with students?

My first important experience involving the teaching of programming style came several years ago when working with a small group of 6th graders. Several of these students began writing long, elaborate graphics procedures. During one of our weekly sessions, one student asked about a program that he was unable to debug. The program consisted of a single procedure that was several screens long. All of the students gathered around the computer as I helped break the procedure into several meaningful subprocedures which were then called by a superprocedure. We found the student's bug almost immediately. He was delighted. One other student in the group remarked "Maybe that's why my programs never run!" That remark coupled with other similar experiences convinced me that even elementary students need some instruction in programming style.

When teaching Logo to high school students, I often encounter what I call a Basic mentality ..."if it runs, it's right". While this approach works when students first begin programming in Logo, it isn't long until workspace are filled with unnecessary statements, unused procedures, and unintentional embedded recursion. I discovered early in my teaching of Logo to high school students that such programs are nearly impossible for students to debug and result in frustration that not infrequently leads to students' deciding that they simply can't learn to program.

When I first started teaching Logo at the high school, I tried to keep in mind the Logo philosophy, and merely made suggestions about style to my students. Few responded. They were too busy getting the program done on time in the midst of their busy lives. They couldn't be bothered with matters of program design. Only when poor design led to nearly impossible bugs were they willing to consider program redesign.

Because of these early experiences with Logo, I have come to feel that teaching, and enforcing, good style is very important when working with beginning programmers. While this may only mean encouraging the use of meaningful procedure names and keeping procedures fairly short with young students, as students become older and/or more sophisticated, it becomes increasingly important to impose more stylistic restrictions designed to guide students toward better programming habits.

In my beginning high school Logo class, I have established a number of guidelines which I find greatly reduce the number frustrating situations which students encounter. Some of these restrictions are admittedly quite artificial. For example, I require that the main or superprocedure consist only of user-written procedure calls. This immediately forces students away from writing one long procedure and leads to more appropriately modular programs. Another somewhat artificial restriction is that procedures should not be "chained"; that is, a program should not be designed such that procedure ONE calls procedure TWO which calls procedure THREE and so forth. This prevents students from simply writing one procedure and breaking it into several procedures by adding TO's, and ENDS here and there.

One of my less arbitrary guidelines focuses on the use of meaningful names. Basic programmers always want to use one or two letter names and even high school students will give procedures silly names. Good naming not only helps the program author debug, but it makes it easier for fellow students to participate in the debugging process, creating a richer learning environment in the computer lab.

Because I encounter so many Basic programmers, I teach MAKE or NAME quite late in the course so that students must learn to use procedure inputs. Once global variables are introduced, I insist that they only be used when it is necessary or "reasonable". In general I find that this makes students think about why they use a particular kind of name and in the process think more about their program as a whole.

My experience to date is that imposing programming guidelines such as these is well worth the effort. It does not seem to impare their creativity, but rather it gives them more powerful tools with which to express themselves. When students grumble, I remind them that there are restrictions placed on how they turn in English papers or science lab reports as well. I usually ask what would happen if they turned in an English paper with no punctuation or capitalization. When they laugh, I point out that programming is not all that different. This particular analogy is quite effective is helping students understand that the guidelines I set for them have a lot to do with writing readable programs.

As my students become more sophisticated programmers, I gradually relax my restrictions on style. Most students evolve their own programming style. While each student's style is unique, most retain some flavor of my guidelines. Few return to writing Logo programs that look like Basic!

The important point is, I believe, not whether we teach style, but how we teach it. There already exists a defacto style for writing Logo. A look at almost any published material should be quite convincing. There is a danger of accepting this style as the way to write Logo. Instead, we need to listen to each other and to our students. We need to rethink our programming style each time a new version of Logo offers new features or more power. We must, I believe, remain open to reasonable and creative ways of writing Logo that still result in programs that are relatively easy to read and debug.

Programming Styles to Meet Different Needs

Marlene Kliman
MIT Media Technology Lab
20 Ames Street
Cambridge, MA 02139

Alison Birch
Terrapin, Inc.
222 Third Street
Càmbridge, MA 02142

Since we teach concepts of good programming style and techniques right at the beginning, why not continue to teach them at more advanced levels, as we learn to use Logo as a more sophisticated tool for thinking about mathematics, language, science, and problem solving? In this talk we will discuss theoretical and practical reasons for selecting a particular programming style. We will be providing a comprehensive handout containing concrete examples of Logo programs and tool procedures.

If we are committed to the premises that one child's program is no "better" than another child's, that the idea of giving tests in Logo is ridiculous, and that we should accept different learning styles for different individuals, then why is it that we cringe every time we see a published Logo program written in "bad" style? Is this a contradiction? Or is it that learning and teaching "good" programming style leads to better thinking skills and a deeper understanding of particular problems and their solutions? Certainly there are many reasons why style does matter. Well-written programs are easier to read, debug, and modify.

Some of the most basic rules of Logo programming style are always taught right from the beginning, they seem to be part of the "Logo philosophy":

- Procedures
- Modular programs
- Meaningful procedure and variable names

And, at an intermediate level,

- Recursion

Usually, the style in which a program is written reflects the way the programmer thinks about that particular problem and its solution. If we are going to use Logo as a tool to help people think about ideas in new ways, then paying attention to good programming style can help.

However, when writing a program, we may have several purposes to fulfill. We may want to write a program that takes up as little disk space as possible, we may want to write a program that is easy to understand and can be used in class with beginners, or we may want to write a program that shows off an efficient but perhaps cryptic algorithm. Sometimes these purposes conflict. For example, a program designed for teaching purposes may use many PRINT commands to help students in understanding and in debugging. Text included in procedures can take up a lot of space, so the inclusion of many PRINT commands might not be appropriate if the program were constrained to fit into a small space.

In our talk, we will discuss several kinds of purposes and constraints - such as programming for teaching purposes, readability, efficiency, and maximizing stack space, node space, and disk space - that might guide choice of programming style. We will discuss a series of programming techniques in light of how appropriate they are for meeting these purposes and constraints.

In particular, using sample programs we will show when and how to use:

* Text files

* Self-erasing procedures

* Modular procedures

* Local and global variables

* Automatic garbage collection

* Variables in recursive procedures

* Conditionals

By teachers from the Lamplighter School Inc.
11611 Inwood Road
Dallas, Texas 75229
1-214-369-9201
Sheila Leventhal
Coleta Lewis
Marty Melton
Theresa Overall

TOTAL IMMERSION, SINK OR SWIM

Where students who lag in their fine motor skill development are hindered in the traditional pencil and paper activities, what about the student who has an aversion to the computer. Is the student with an aversion to using the computer hindered in a totally computer curriculum? What are the causes of this aversion and how can it be overcome or compensated?

At the Lamplighter School, there are fifty one computers for students' use and 475 students. All students use Logo exclusively except for a word processor. This ratio of computers to students allows the opportunity to extend traditional curriculum and create new environments of learning by use of high technology. The teaching staff has attempted to extend all areas of the curriculum through use of the computer by the students. This effort is called "Total Immersion".

The students at Lamplighter range in ages from three years old to ten years old. At the fourth grade level, three teachers have extended curriculum or created new microworlds where students could learn by doing or by manipulating objects in the world around them. The following paragraphs are some of the ways these teachers have done this at the fourth grade level.

1. MATH

   A good review and application of subtraction skill is "balance and payment charts" where students use repeated subtraction to calculate standing balances and to see how many payments it will take to pay off a total balance. The same balance and payment is an excellent introduction of or application of variables and recursion with variables.

   Dot is a concrete introduction of x and y coordinates. After students draw pictures with DOT in the first quadrant, they are introduction to the other 3 quadrants and then asked to "stretch" their pictures taller or wider by changing one or the other coordinate.

2. SOCIAL STUDIES

Objective: Students will be able to program using both graphics and text on the same screen. The final program may include as many different scenes or "screens" as they like, but should have a minimum of two scences with different text for each. Some students may develop a complete story with illustrations.

Source: The theme for the stories or reports is focused on the Texas unit. They may create a story or report with pictures depicting something which interests them about Texas.

Set-up: Students will work in pairs alternating time at the keyboard and time giving ideas and oral assistance.

3. SCIENCE

Field of vision

Objectives:

1. Describe how field of vision (area) depends on distance of field from vision.
2. Construct a graph to show relationship between field of vision and distance.

Rationale: As a person moves away from an object, the person is able to see a larger area of the object. The field of vision increases as the distance between the item being viewed and the viewer becomes greater, and correspondingly decreases as this distance becomes less.

After experimentation the children will observe how field of vision changes as distance changes. Observations will be interpreted from a generalization about the relationship between distance and field of vision.

Activities: Activities require children to record changes in field of vision corresponding to changes in distance from object. Children are asked to quantitatively relate both linear dimensions and area of field of vision to distance from object.

Children use graphs to summarize observations, to predict and to generalize.

Upon completing their observations the children then translate their findings to a data storage table (or graph) using the computer.

Children are then asked to develop procedures which would graphically and clearly show the generalization and predictions they formed about the relationship between distance and field of vision.

4. Others: Other areas of the curriculum will be presented in less detail such as creative writing, and language activities.

With a program of "total immersion" in the elementary school curriculum by using the computer lanauge Logo to extend or create new microworlds of learning, is there a danger of too much technology? Naisbitt, in Megatrends, states that we develop our own ways to compensate for the high-tech influence of the computer. Do fourth grade students develop ways to compensate for high-tech? Is Logo, when used in an open-ended discovery method of learning, an element of the high-tech Naisbitt refers to? If high-touch personalities find ways to compensate for the imbalance between high-touch/high-tech, what are they? If computers are used by all students in every aspect of the elementary school curriculum is it a question of "sink or swim?" These and related questions are observed in the classroom and will be presented with this report.

Method of report: The findings and the total immersion curriculum will be presented by use of:

1. Video tape
2. Slide presentation
3. Overheads
4. Oral report

For more information contact Coleta Lewis, computer coordinator, Lamplighter School Inc., Dallas, Texas.

Alexander Goldowsky
Computer Coordinator:
The Phoenix School;
The Palfrey Street School

WORKSHOP PROPOSAL: M.I.T. LOGO '86 Conference

EXPLORATORY SOFTWARE FOR A GLOBAL STUDIES CURRICULUM:
From Logo to Logo Software

Introduction:

Looking at software developed for a world geography and mapping curriculum at the Phoenix School in Cambridge, we will discuss how Logo ideals, including exploration and user control, can be extended to thematic software written in Logo.

Abstract:

In this session we eill work with several programs designed to teach mapping skills, give students experience reading and following maps, and tools to allow students todraw maps to different representational scales (i.e. population density, resource use, etc.). The programs are being used as aprt of a year long global studies curriculum (k-7) at the Phoenix School (Cambridge, MA). The curriculum integrates not only Social Studies and Science work, bur forms a unifying them for students' classes in Art, Language Arts, and other subjects.

In designing software and computer activities for this curriculum, the challenge has been to write thematic programs that build on the Logo philosophy, including an emphasis on cooperation, self direction, user-control, problem solving and experiential education.

I will demonstrate several programs, using them as a jumping off point for a discussion of the following issues:

*How software, or tool activities, can be designed to allow students to explore a topic such as geography, a subject that has taditionally been the province of drill and practice CAI programs.

** How to present curriculum material as problems, rather than presenting answers or facts.

** How the internal structure of a program can model the actual phenomenon and thus be far more flexible and accurate.

** How programs can be written to allow students to use their knowledge of Logo to extend and personalize software, and how this can lead students to a greater formal understanding of the topic, as well as to a clearer sense of the uses and limitations of computers.

In looking at the mapping software I hope to present both an example of how Logo software and activities can be integrated into a social studies curriculum, and a look at some of the issues inherent in the design of such software. Another emphasis will be on the advantages the structure of Logo affords in the programming of such exploratory software.

Workshop Outline

Introductions, and outline of the structure of the school and the interdisciplinary global studies curriculum. Quick description ot the computer component of the curriculum.

Explaining and playing the cooperative mapping game "Rover." Throughout, the programs will be presented with an eye towards their role in the curriculum, and the design issues involved.

Demonstration of the scale mapping tools, and some of the work students have done using them.

Discussion; tying together the issues and generalizing.

Additional time for trying out programs as the length of the seminar and equipment permit.

Equipment

Two Apple 2E's with Terapin 3.0 Logo, and extended video cables; one dimension monitor, or multiple monitors, attached to one of the machines.

Design and Development of a Comprehensive, Structured, LOGO Curriculum Manual for the Elementary School: Grades Kindergarten - Five

Judith W. Weinberg and Nancy J. Fralic

Pittsburgh Public School District
Pittsburgh, PA

In an attempt to integrate LOGO into the elementary school curriculum, a structured LOGO curriculum manual with accompanying software has been developed by Pittsburgh Public School District teachers and supervisory personnel. The Manual includes the rationale for incorporating LOGO into the elementary school, a scope and sequence of objectives, a user's guide for the hardware and software operations, LOGO references and resources, and a set of detailed lessons for each grade level, kindergarten through five.

The District has provided extensive on-the-job training and after-school and Saturday inservice sessions for its teaching personnel. However, teachers expressed the need for an additional resource that would help them utilize the LOGO language within the framework of District-based instructional objectives. Thus, a comprehensive curriculum has been created that not only teaches literacy and LOGO programming skills but also integrates the teaching of LOGO into several curricular areas. The LOGO language is used, therefore, as a tool to teach District-based objectives in the areas of visual arts, mathematics, science, language arts, perceptual and social interaction skills.

This presentation will include a slide/tape program which will highlight key aspects of the Pittsburgh Elementary LOGO curriculum and its integration into the elementary school program. In addition, the Pittsburgh Public School District's LOGO Curriculum Manual, the accompanying lesson-specific software, as well as student projects will be demonstrated.

The District is now expanding the LOGO program into the middle school. District teachers are in the process of developing LOGO investigations that teach to middle school level instructional objectives in mathematics, visual arts, and science. These investigations will be completed by June and will also be shared at the conference.

Dr. Judith Weinberg is Supervisory Instructional Specialist for Computers in Education.

Mrs. Nancy Fralic is a teacher at Brookline Elementary School.

MusicLand

Applications in Music Using Logo

Developed by:
James Fry
Novi Community Schools, Novi, Michigan
Austia Hollen
Southfield Public Schools, Southfield, Michigan
Lary Smith
Wayne County Intermediate School District, Wayne County, Michigan

MusicLand is a microworld created to help explore the elements of music and sound through the Logo language. The **MusicLand** microworld provides some structured music activities or games along with a set of "tool" procedures to explore and discover many areas of music.

MusicLand was developed as part of a two county Logo integration project. The goals of the project were to develop Logo activities that could be used by classroom teachers, with little or no Logo experience, in the different curriculum areas. One of the groups that formed was a music group. This is the group that developed the **MusicLand** Microworld.

There are two major parts to **MusicLand**. The first is a more structured set of activities called Music.Games. Music.Games consists of 3 different "games"; **High/Low, Long/Short, and Up/Down**. These games are aimed at the Kindergarten to second grade levels. The objectives for the games are to distinguish between high and low pitches, long and short tones, and upward and downward movement of notes in a musical line.

The second part of **MusicLand** consists of a set of "tools" grouped into **"write.tools" and "sing.tools"**. These "tools" have a set of activities that are much more open-ended. The tools allow you to write a tuneblock, sing it (hear it played by the computer), and manipulate it many different ways or combine many tuneblocks into a song. The many things that can be preformed on a tune block are; **Lengthen or Shorten** (the use of augmentation and diminuation), **Reverse and Mirror** (play backwards or invert vertically around a tonic or home tone), **Transpose** (to raise or lower by steps or octaves). These "tools" can also be combined to create many new sounds from one original tuneblock.

All the **MusicLand** activities are structured in the same way to give a beginning user of Logo or the **MusicLand** microworlds a guide to follow. Each lesson consists of the following major topics; Topic presentation, Pre-computer Lesson, Guided Discovery/Logo Activity, Follow-up Off-computer Activity, Extended Learning Activity, and Notes to the Teacher.

THE MUSIC LOGO LANGUAGE

Karen M. Moran
National College of Education

Methods, insights, and conclusions from exploring the recently announced Music Logo language for three months with a normal fifth grade class will be presented. Developed by Jeanne Bamberger of M.I.T. and Terrapin, Inc., Music Logo is the same as Logo except that the graphics primitives have been replaced by music primitives. In addition, Music Logo requires a nine voice music synthesizer, amplifier, and stereo speaker system.

Whereas graphics Logo provides a mathematical learning environment, Music Logo provides a music learning environment and lends itself to almost endless musical projects. For example, a CANONS command allows the student to compose and experiment with a four part canon. Another procedure provides the tools for a child to discover the effects of different basses (e.g. cowboy, blues, Oriental) on a melody of his or her own creation.

The most significant result of this three month experiment with Music Logo was that normal fifth grade students became composers at a level usually enjoyed only by college music students. With the power of the computer to assist them, the students created and listened to their own polyphonic compositions (canons), and their own ostinato bass compositions (soft rock). The children composed melodies in ABA form, and explored (again by way of the power of the computer and their own programming) the effects of style, tone color, rhythm, and harmony. In short, Music Logo supported the children as they built their own musical intellectual spheres.

In contrast, even children who have had private music lessons are rarely taught either composition or improvisation. Hence, private music students had no advantage in the learning of Music Logo. Furthermore, Music Logo does not use standard music notation and no performance skills are required; therefore, Music Logo allows every child to become a composer and to critically listen to his own music. (It was noted that in a PTA sponsored national contest, the only original music composition entries from this elementary school of 430 students were from the Music Logo classroom.)

Another important advantage of Music Logo is that two normally diverse curriculum areas, music and computer programming, can be simultaneously explored. Of prime importance is that Music Logo is a procedural language; hence, structured programming is being experienced at an early age. Students use up to three levels of nested procedures; they also learn file storage and retrieval, editing, booting, and computer terminology.

As stated by Seymour Papert in Mindstorms: Children, Computers and Powerful Ideas, "the computer has the ability to change our relation to other kinds of learning" (p. 47).

Because of Music Logo, the computer is now changing our relation even to the learning of music.

For further information, please contact:

Karen M. Moran
96 Hillcrest Avenue
Glen Ellyn, IL 60137

Gary S. Stager

Paper Proposal: Logo '86

Logo and Music - A Powerful Tool for Learning

Most Logo users have not explored Logo's music potential and many music teachers are not aware of how Logo can be used to enhance their students' understanding of musical issues. It is true that the marriage of Logo and music may provide Logo programmers with new challenges and is an excellent instructional tool for teaching components of the traditional music curriculum. These uses, however, fall dramatically short of the numerous powerful ideas learned by children exploring with music and Logo.

In my two years of experience working with elementary school students, Logo, and music I have observed the emergence of much deeper learning than that of discrete musical theory topics or Logo syntax. In fact, the type of learning that takes place is interdisciplinary. Jeanne Bamberger's research at M.I.T. of the 1970's taught us that a child's use of Logo and music mirrors the learning process itself. In this research we see that Logo and music can help bridge the gap between thinking formally and figurally. If we observe children's interaction with Logo and music we gain insight into individual learning styles.

During my presentation I will discuss some of the areas in which children grow intellectually as a result of their experiences with music and Logo. The following is an overview of these areas.

Problem Solving:

This vastly over-used term can be used to describe many of the issues that evolve during Logo/music use. Logo and music encourages the process of procedural thinking by breaking a problem down into it's component parts and natural learning takes place through exploration and experience. This environment provides many situations in which the student is forced to solve new problems against a pre-existing model and in this process of testing the constraints of an existing model, new concepts evolve. The student's perception of himself/herself as a learner is enriched through the realization that new models can be created and new problems solved.

Thinking Mathematically:

It is no secret that there are mathematical principles underlying numerous musical issues. Many students find music more interesting than math and this provides us with an opportunity to get children excited and involved in mathematics. Learning mathematics through music and Logo helps to dispel the myth that math has to be "hard" and isolated from other curriculum areas.

Logo and music provides students with a new domain in which to explore numbers in new and signifigant ways. I have observed children discovering and learning the following mathematical concepts: counting, addition, subtraction, seriation, symmetry, similarity, transposition, positive/negative integers, decimals, seriation, time, velocity, randomness, and recursion.

Aural Development:

When a student works with music and Logo they must concentrate and listen carefully, in the process their listening and concentration skills improve.

Aesthetic Development:

When a student experiences the process of personal creation they take great pride in their creation. This personal sense of creative accomplishment leads to a greater appreciation of other's work and art that is outside of our own experiences and norms.

Learning Logo:

Music and Logo can be used as an ideal introduction to Logo programming or used to spice up existing Logo activities. Several fundamental Logo topics are utilized when working with music. These include: procedural writing, inputs, commands, operations, the editor, and list processing.

Learning Musical Issues:

Music is probably our most abstract art form and yet we impose a complex theoretical system on it. There is a large body of research that suggests that elementary school age children are not capable of processing this complex system. Logo provides the student with a microworld in which they can gain an understanding of pitch, duration, rhythm, melody, intervals, structure, and notation by actively exploring these topics, at their own pace.

Logo as a Compositional Tool:

Logo and music can be used as a tool for musical composition, regardless of the user's musical experience. Logo provides the opportunity for easy manipulation and editing of musical ideas. Beginners can create elaborate structures with even the simplest tune blocks and more adventureous composers can experiment with complex devices found in serial music.

Conclusion:

There are two types of Logo environments that will be explored in this presentation. The first type is known as tune blocks. Tune blocks are chunks of a song which the user manipulates (as they would with blocks or Lego) to build musical compositions. The use of tune blocks has profound implications for understanding what a "tune" is. In the process of understanding what a tune is, the student not only discovers numerous musical issues, but also develops intellectually in the ways mentioned above.

The other type of Logo/music environment explored in this presentation is that of the "music tool". The music tool can either be provided for the child or created by the Logo user. These music tools are Logo procedures that are used to create sound-effects, manipulate musical ideas, or create graphic notation. All of the tune blocks were created with music tools and therefore students may create their own tune blocks, as well as original compositions. All of the tune blocks and music tools work at TOPLEVEL and can be used with graphics, text, and other music procedures.

I have many Logo/music procedures written for Apple Logo I, Apple Logo II, IBM Logo, and LCSI's Macintosh Logo. Of course, I will be glad to share theses materials with conference participants.

Respectfully Submitted:
Gary S. Stager
Network for Action in Microcomputer Education
12 Locust Place
Wayne, N.J. 07470
(201) 831-0133

Logo and Math on the Mac

Lynda Colgan

Lists. We hurriedly write "milk" on the grocery list stuck to the refrigerator as we drink that last refreshing drop. Teachers frequently record the list of homework assignments on the blackboard at the front of the room. Students scribble "To Do" lists inside their cluttered locker door. LOGO-using educators are challenged to complete a simple interactive procedure that will create Haiku poetry using LOGO and program the computer with lists of nouns, adjective phrases and adverbs. Lists can take many forms, but in the final analysis they are all the same: tallies.

And what of the tally summarizing LOGO in the secondary math program? On the "dream list" we find that LOGO makes the following promises:

- To be a vehicle for the encouragement of individual, independent, discovery-based learning.
- To contribute to a classroom environment that fosters curiosity and a hunger for learning.
- To promote higher order thinking skills such as analytic ability and interpretive skills.
- To maximize communication skills via co-operative and conversational learning.
- To exemplify process education.

On the tally that measures the "nuts and bolts" reality of the classroom, we find that teachers continue to record these concerns:

- Large classes comprised of students of widely varying abilities.
- Voluminous curriculum documents and concomitantly, time constraints.
- Accountability. (Justification of program; Evaluation of students.)
- Limited hardware.
- Scarcity of appropriate In-Service programs.

And the tally that summarizes much of LOGO's impact on the classroom to date? KEYSTROKES instead of MINDSTROKES.

The challenge? To record a new tally in response to marrying the dreams to the realities and LOGO to Math in which the grand total reads, "here is a new view of computers and education."

Four Macintosh microcomputer systems were donated by the Apple Education Foundation to facilitate a local mini-project in which secondary teachers and students will be encouraged to explore LOGO and concomitantly be challenged to take a different look at the mathematics curriculum. In this project, teacher and student alike will participate fully in the very active role of learning partner. Along with a "computer coach", the teachers and students will form a team involved in the process of problem solving, the creation of new investigations and the design of extensions.

Why was the Macintosh chosen? Elementary school math teachers employ wonderful **concrete teaching aids** - Cuisenaire rods, geoboards, construction paper and scissors, miras, dot paper, pipe cleaners, cards, films, and people! Children perform experiments, act in skits, and view films all related to a key concept, such as division. The activity-centred, student-oriented classroom environment reflects the credo of the elementary teacher, who explains that the principle objective is to "teach children, not mathematics." Secondary school math teachers, as a rule do not use manipulative aids. On one side of the coin, they explain that students react negatively to "kid stuff" and on the other, that there is little time for discovery-learning because of short periods and heavy curriculum demands. The solution? The Macintosh. A very cleverly disguised and extremely sophisticated set of manipulative materials including scissors, grid, calculator, markers, ruler and protractor. An entire mathematics laboratory rolled into one! In addition, it was felt that the user **interface** of the Macintosh would reduce user anxieties (secondary school teachers and students can be surprisingly intimidated by the technology) since it's forgiving, "I wonder what if?" nature invites playfulness and experimentation. Furthermore, it was felt that the Macintosh will be highly motivational since it is a **new** tool for most of the teachers and students, and at the same time will be more **transparent** because of its state of the art nature, making most tasks highly accessible. In addition neither teacher nor student will have a preconceived association of this microcomputer with traditional C.A.I. math applications, leaving the door open to new types of computer-based learning activities.

Why LOGO? First and foremost, the underlying **philosophy** of LOGO is in discovery learning. Discovery learning must not be designated as an elementary-school teaching strategy. Discovery learning challenges students to go beyond the facts, to make sense of seemingly ambiguous or contradictory information, and to organize and reorganize information in new ways that are not immediately apparent to the eye. Many math classes are structured in such a way that the teacher is the keeper of the knowledge. Homework is discussed. A new lesson is taught. Seat work is assigned. An extension of the lesson is presented. Homework is assigned. The class is dismissed. Because the classroom is a centre of teaching as opposed to one of learning, students never develop a sense **of ownership of an idea** or concept. In student-centred discoveries, the emphasis is on the learner actively seeking resource information, bringing together nuggets of ideas from many areas, and organizing the knowledge in new ways in order to hypothesize and test a possible solution. With each stage, comes a stronger bond of ownership of the newly discovered idea. And after all, we like no ideas better than our own, do we?

In addition LOGO's **extensibility and procedural organization** most certainly contribute to the learner identifying his personal procedure definitions and designing a manageable collection of thinking and problem-solving strategies. Since there is an appreciation of "belonging to" not only the task but the approach used in organizing and solving the **problem**, there should be some **transfer** of strategy to other subject areas. A Computer Resource teacher in the North York Board of Education, and President of ECOO's SIG-LOGO, Peter Skillen recalls the following powerful classroom incident as an illustration of the powerful role played by LOGO in the transfer of a concept:

> **Our grade two/three class had the opportunity to watch the overpass of the space shuttle piggybacked on a jumbo jet. When we returned to class, a discussion about space naturally erupted. One child asked if Earth was in space, and in asking the question, we determined that yes, it must be, because it wasn't sitting on anything. The discussion continued until Jeffrey piped up.**
>
> **"You know it's sort of like LOGO."**
>
> **The class stopped, and looked at him curiously, as I did myself.**
>
> **"What do you mean?" I asked him curiously.**
>
> **"Well, Earth is like a procedure. It's like a sub-procedure inside the solar system. The solar system is the super-procedure and the solar system is like a sub-procedure inside the Universe. The Universe is like the Super-procedure."**

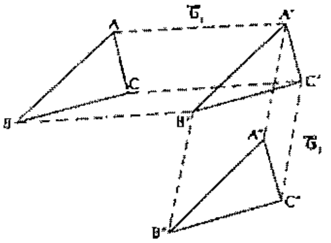
Appreciating the power that LOGO's procedural base has in contributing to transfers such as these at the elementary school level, secondary school teachers should be encouraged to employ this programming language in their classrooms in order to promote older students' looking at situations in new ways.

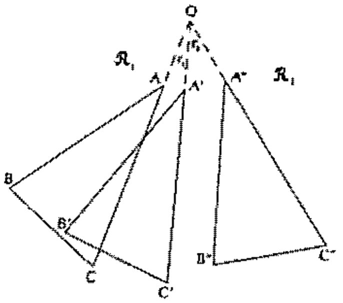
Thirdly, LOGO on the Macintosh is extremely powerful. It can be **tailored** to meet the needs of the user by customizing primitive sets and memory preferences to accomodate graphics investigations, list processing applications or a blend of the two. It has other improved features such as natural logarithms, a debugging menu, dribble file capabilities, and user-friendly "cut and paste" editing features and print commands.

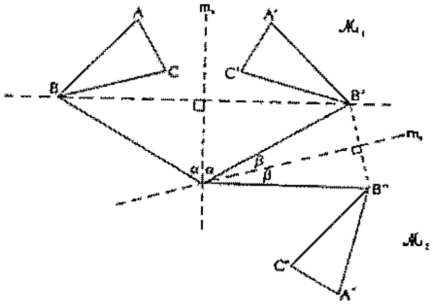
In the first stages of the project, the teacher-student partners will be challenged to use simple MacPaint activities to create a variety of polygons and then using the pull-down windows, edit the shapes to explore translations, reflections, rotations and symmetry. MacPaint will allow the students to access the task quickly, and in original, creative ways. Teaching aids such as acetate rulers and protractors, tracing paper and hardcopy will be used in the discovery explorations, and problems will be fully discussed in the spirit of teamwork.

Although the MacPaint program is powerful enough to create visual models of many of the principles of transformational geometry, the students will soon discover that they have no control over such factors as exact co-ordinate placement or axis incrementation, nor are they creating a truly dynamic model of the transformation process. It is at this point that LOGO will be introduced as the tool that is needed to explore these concepts to their fullest.

After an appropriate opportunity to explore the simple primitives and editing features of LOGO, the teams will be challenged to work co-operatively to design a super-procedure that will assist in the exploration of the products of two or more transformations (e.g. What is the product of two translations? two rotations? two reflections?). After creating the first procedures, LOGO can be used to "What if?" extensively. Teams can experiment with mirrors that are parallel or mirrors that intersect. These exercise, although relatively simple and certainly standard in many LOGO math applications, will be used as the springboard, i.e. the introductory exercise in which every student is assured some degree of success and that is used to build confidence and competence. As well, the explorations will have accomodated many types of learners - those who require a structure, those who use top-down problem-solving strategies, those who employ bottom-up techniques, and those who demand to be challenged beyond the visible limits of the task.

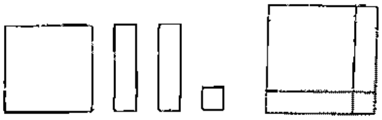




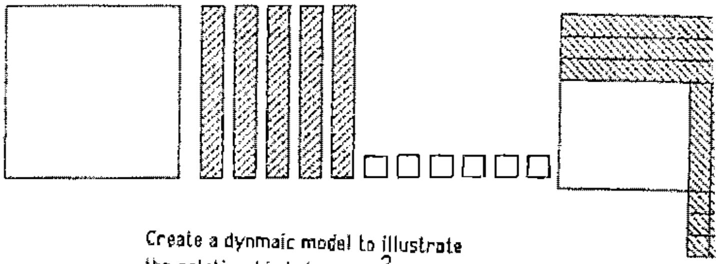


Once the learning teams have built their personal microworld for exploring transformations, they will be challenged to explore new concepts, such as factoring trinomials, but with a difference! Too often students memorize formulae and apply "fill in the blank" recipes in order to arrive at the correct response without really appreciating that a formula, for example is merely one representation of a mathematical idea. This is most evident, as a class completes every chapter with the predictable WORD PROBLEM section. Unfortunately, for most students, word problem extensions are not perceived as a natural extension of a particular unit, but rather are viewed as a topic divorced from the main theme.

In this portion of the project, students will be challenged to put their LOGO knowledge to use and inherently extend their mathematical knowledge by constructing visual models of trinomials in order to build a general algorithm and an interchangeable algebraic/geometric understanding of the concept. This is diametrically opposed to the more traditional classroom lesson in which students are asked to solve (destruct?) a series of such problems using the traditional "recipe" approach. By creating a dynamic LOGO model, reality and significance will be added to expressions such as $x^2 - 5x + 6$. Square of side x, rectangles and unit squares are physically added and removed from the model, with the learners determining the appropriate operations.



Create a dynamic model to illustrate the relationship between $x^2 + 2x + 1$ and $(x + 1)^2$.



Create a dynmaic model to illustrate the relationship between $x^2 - 5x + 6$ and $(x - 2)*(x - 3)$.

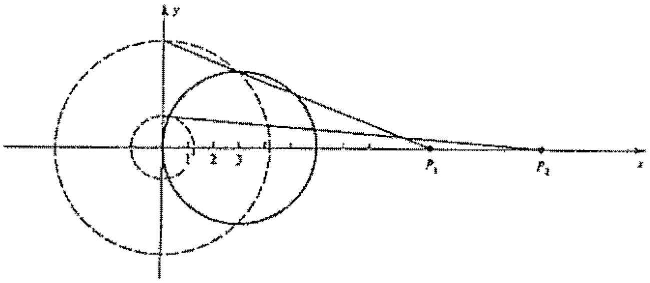
As E. Paul Goldenberg reminds us in his book Computers, Education and Special Needs, "learning to calculate is not an adequate objective for school mathematics." When students are afforded the opportunity of learning LOGO and through LOGO learning about math, they will recognize a formula as having interrelated components and at a later point should be able to rely upon visual models to determine whether or not an algebraically determined solution is reasonable. If then the learner is confronted with a word problem that reduces to an equation such as $x^2 - 7x + 12$, the task of drawing a diagram will not seem to be an "add-on" to please the teacher as part of the mark allocation in the "Rules to Follow For Word Problems" recipe, but rather a natural step in deciding upon and testing a reasonable answer – a far cry from computing the answer!

As an extension, simple building-block procedures using squares of side **x**, rectangles and unit squares could be employed to build dynamic and visual models of the very obstract notion of "completing the square." Grade twelve textbooks continue to introduce and discuss this very troublesome concept exclusively in algebraic terms. When asked "What must we add to $x^2 - 6x$ to make it a perfect square?" many students and teachers do not have a visual mathematical model upon which to rely. If the learner builds procedures to explore geometric models of expressions such as "Complete the square for: $x^2 - 8x + 13$ and $x^2 + 10x + 12$", then future activities are not reduced to a series of steps applied to both sides of an equation, for example, but are meaningful. The learner has proposed conclusions about the interchangeable algebraic/geometric nature of such expressions, is continually testing the reasonableness of those conclusions, and will be challenged to apply these newly-discovered concepts to extensions and new situations.

The following problem, posed by Johnson and Rising, in their teacher education book Guidelines for Teaching Mathematics outline the following problem.

> A fixed circle of radius 3 has its centre at the point (3,0) on the x-axis. A second circle has its centre at the origin. Its radius is made to approach zero. Consider the line joining two points on this second circle: The intersection with the y-axis and the intersection with the other circle. How far from the origin will this line intersect the x-axis?

At first glance, it may appear to draw together many seemingly unrelated parts of mathematics, and yet, as we study the problem under the numerical microscope, we recognize that its solution provides the learner with an insight into the process of limits. As the learner creates this model in dynamic LOGO form, there will be the discovery that as the variable circle shrinks, the points of intersection of the line with the x-axis move further to the right; however, even as the procedure is modified, there will be the discovery that the line does not move to the right an unbounded distance. No matter how small the radius of the variable circle becomes, the point of intersection of the line and the x-axis cannot exceed 12.



Once more, it will be impossible for the learner to conclude that the teacher "fixed" the example, and demonstrated a diagram of a one-time phenomenon, since the visual model was built, refined, tested and applied by the learner alone.

There are battles to be fought, however. Why?

First of all, there are still the realities of the classroom such as large numbers of students, voluminous curriculum, limited hardware and student evaluation. This schema does not provide the answers to those problems. What the schema does address is the more global issue of implementing the process of change in the traditional approach to math teaching. However, the teacher is not abandoned, because the introduction to LOGO will emphasis pedagogical strategies such as: tailoring investigations not only to meet the needs of a variety of learners but to satisfy the course requirements; including group projects and conversational evaluation techniques; and sharing of hardware and knowledge resources with other staff members. The Jig-Saw approach enacted!

Secondly, teachers are traditionally reluctant to become co-learners. The dynamics of the classroom environment will change absolutely in the implementation of such a LOGO project; however, since the introductory sessions will be conducted using a learning-team approach, teacher anxieties should be reduced. The teacher will have first-hand experience (in a non-threatening environment) of his/her emotional response in shifting the locus of control to the student. No longer will the teacher simply tell students things. The role will be of master poser of challenges and questions. But how does one deal with the unanticipated questions, the tangential explorations, and the accumulation of resources? How does one deal with looking at familiar questions through different eyes? There are some answers to the questions, however, in that through the introductory sessions, the teacher will have been guided in the development of a strategy for full implementation of LOGO-based math units and built a framework upon which to extend the philosophy and practice of such program to a full class of students.

Next, most teachers and students anticipate computer-based math applications to be C.A.I. activities. They will not immediately recognize the appropriateness of programming a computer in a math class, nor will they instantly appreciate the efforts that must be taken to collect resources and design explorations in so open a computer environment.

Lastly, this project is not suggesting that LOGO be integrated into the curriculum, but that it is possible to base sections of the curriculum on LOGO investigations. In other words, the hypothesis is that through focused challenges, LOGO can teach key mathematical concepts, and still preserve the notion of discovery.

In conclusion, we return to the notion of lists. Lists of teacher names awaiting In-Service training. Lists of support mechanisms for classroom computer-using educators. Lists of strategies for making LOGO work in the classroom. Lists of LOGO challenges that can teach mathematical notions. Lists of questions to be asked in arriving at a possible solution. Lists of resource information. Lists of procedures. Lists of discoveries. Lists of extensions.

There is much to be done to see this project come to fruition in order that the concrete materials generated can be shared with classroom teachers. And what will guarantee its completion? Lists of dreams.

I would like to thank the Apple Education Foundation for so kindly awarding me the necessary hardware and software for building a "To Be Shared" collection of challenges and strategies for actively employing LOGO in the secondary math classroom. I would also like to apologize to the Foundation for being slightly behind schedule.

I would also like to sincerely thank Pat Rogers, jointly appointed to the Mathematics Department and Faculty of Education at YORK University for her time, creative ideas (Trinomials & Completing the Square), skillful questions, enthusiastic commitment to quality education, and knowledge of the language of the turtle!

LOGO AND LINEAR ALGEBRA

One of the best ways to learn a new concept is to teach it to someone else. We are taking an independent study course in which we have found this to be true. One of the projects we have been working on is transforming basic linear algebra algorithms into Logo. Logo is a new concept for us. By engaging in this project we have both been sharpening our linear algebra skills (by teaching linear algebra to the computer) and learning a new powerful computer language.

The first major problem we come across is how to represent vectors and matrices (and distinguish between the two) using Logo. This is solved by using lists.

The vector (1,2,3) is represented by the list [1 2 3].

The matrix $\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$ is represented by the list [[1 2 3][4 5 6][7 8 9]].

It is important to be able to recognize the differences between vectors and matrices. For example:

[1 2 3] is a vector while [[1 2 3]] is a one-by-three matrix, and

[[1] [2] [3]] is a three-by-one matrix.

As we learn more about Logo and linear algebra, it becomes easier to write basic procedures combining the two. It is very simple to transform vector algebra formulas into Logo procedures once we are able to perform the three basic operations on vectors (addition, scalar multiplication, and dot product). We call these three basic operations ADDV, SCALEV and DOTP.

Our program DOTP uses Logo primitives. It is as follows:

```
TO DOTP :VECTOR.A :VECTOR.B
  IF COUNT :VECTOR.A = 1
              [OUTPUT (FIRST :VECTOR.A ) * (FIRST :VECTOR.B)]
  OUTPUT
    (FIRST :VECTOR.A) * (FIRST :VECTOR.B)
            + DOTP (BUTFIRST :VECTOR.A) (BUTFIRST :VECTOR.B)
END
```

Now that this is accomplished, we are able to define component (COMP) much like we would in linear algebra. In linear algebra, component is defined to be the dot product of vector A and vector B divided by the dot product of vector B and itself. In our Logo procedure component is as follows:

```
TO COMP :VECTOR.A :VECTOR.B
  OUTPUT (DOTP :VECTOR.A :VECTOR.B) / (DOTP :VECTOR.B :VECTOR.B)
END
```

Using these simple procedures, we can build other, more complex procedures. For example, in linear algebra, if A and B are vectors, the projection of A on B is defined to be the scalar product of B by the component of A on B. Our Logo procedure looks like this:

```
TO PROJ :VECTOR.A :VECTOR.B
  OUTPUT SCALEV (COMP :VECTOR.A :VECTOR.B) :VECTOR.B
END
```

Once vectors are easily manipulated, the operations on matrices soon follow with little difficulty. The procedures dealing with matrices are written by applying the vector procedures recursively to the rows of the matrices. For example ADDM is defined in terms of ADDV:

```
TO ADDM :MATRIX.A :MATRIX.B
  IF :MATRIX.A = [] [OUTPUT []]
  OUTPUT FPUT (ADDV FIRST :MATRIX.A FIRST :MATRIX.B)
              (ADDM BUTFIRST :MATRIX.A BUTFIRST :MATRIX.B)
END
```

In the same manner, SCALEM can be defined in terms of SCALEV. However, multiplying matrices is a bit more difficult. In linear algebra, matrix multiplication is defined as follows: if A and B are matrices, and A is m x n and B is n x p, then AB is m x p and the i,j entry of AB is the dot product of the ith row of A and the jth column of B. For example:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 0 & 1 \end{pmatrix} \begin{pmatrix} 2 & 1 \\ 3 & 6 \\ 7 & 2 \end{pmatrix} = \begin{pmatrix} 29 & 19 \\ 15 & 6 \end{pmatrix}$$

Because we represent matrices as lists of rows, it is difficult to multiply a row of one matrix by a column of another matrix. To remedy this problem, we have developed a procedure to take the transpose of a matrix. Transpose takes the i,j entry of one matrix and turns it into the j,i entry of another matrix. For example:

$$ {}^{t}\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} = \begin{pmatrix} 1 & 4 & 7 \\ 2 & 5 & 8 \\ 3 & 6 & 9 \end{pmatrix} $$

The transpose of B is taken in our procedure MULTM (by taking the transpose in a separate procedure, time is saved because the transpose is only taken once instead of each time the transpose is needed). Our procedure looks like this:

```
TO MULTM :MATRIX.A :MATRIX.B
  LOCAL "T TRANSPOSE :MATRIX.B
  OP MULTMM :MATRIX.A :T
END
```

Our procedure MULTVM takes the dot product of any vector A with each row of a matrix T. MULTMM uses recursion to take the dot product of each row of a matrix A with each row of a matrix T. They are as follows:

```
TO MULTMM :MATRIX.A :T
  IF :MATRIX.A = [] [ OUTPUT [] ]
  OUTPUT FPUT MULTVM FIRST :MATRIX.A :T MULTMM BUTFIRST :MATRIX.A :T
END

TO MULTVM :VECTOR.A :T
  IF :T = [] [ OUTPUT [] ]
  OUTPUT FPUT DOTP :VECTOR.A FIRST :T MULTVM :VECTOR.A BUTFIRST :T
END
```

Our talk will elaborate on the preceding topics and will also include a discussion of other procedures that we have worked on. Some of these procedures calculate the cross product of two vectors, the angle between two vectors, the equation of the plane containing three points, and the length of a vector.

For the remainder of this school year, we have several projects in mind. We plan to: work on a package which will use Logo graphics to illustrate linear transformations of the plane, write a routine that will reduce a matrix to echelon form, and write a general determinant procedure.

We feel that by working on this project we have accomplished a number of things. Because of the procedural aspects of Logo, it is easy to understand how linear algebra is based on basic algorithms which can be called to do more complex problems. Also, while we have been reviewing linear algebra and learning Logo, we have been creating a useful package for future linear algebra classes. Finally, we have reinforced the theory that teaching something to someone (in this case, the computer) strengthens the teacher's grasp of the subject matter at hand.

Kenneth E. Bouley
Stephanie L. Ragucci

c/o Albert Cuoco
Mathematics Dept.
Woburn High School
88 Montvale Ave.
Woburn, MA 01801

Naomi Bolotin

THE SUMMERMATH PROGRAM: A CASE STUDY IN USING LOGO

One of the wonderful features of a language like Logo is that one can use it as a tool to help teach other subject matter. An example of this is a program that runs for six weeks every summer at Mount Holyoke College. Called Summermath, this program is designed to help high school girls who have had trouble with mathematics learn to understand, enjoy, and appreciate the subject. The program is now starting its fifth summer, and since 1982, over three hundred and fifty girls have attended the program, coming from as far away as Alaska, the Dominican Republic, Nepal, and West Germany. Most of the girls come from other parts of the country and thus live in a dormitory on campus, getting their first taste of what college life is like. In addition, there are several local non-residential students who attend each summer. The girls have three ninety-minute math classes each day Monday through Friday. The classes are a Fundamental Math Concepts class, a series of three two-week math workshops (which the girls select), and a Logo computer class. In addition, they take recreation, arts and crafts, and creative writing classes, go on trips on weekends, and participate in various other activities in the residential life part of the program.

The purpose of the computer class is to give the girls a chance to use the mathematics that they have learned previously by applying it to a variety of interesting math-related projects. Each class consists of approximately twelve students, who work either in pairs or alone, and one or two teachers. All the classes cover the same basic core of Logo material and mathematics, although the method of presentation and choice of activities is left to the individual instructors. This enables us to tailor the material more to the specific needs of the group we have. Since many more girls are now coming in with previous computer experience, and since their mathematical backgrounds vary as well, this summer we are planning to group the students according to their programming and math experience. This will enable us to focus the projects even more closely on the specific material that they have had.

The format we use is an informal lecture/lab arrangement. If we want to present a new programming concept, mathematical topic, or activity for the girls to do, we do this as a group. We illustrate the material using the blackboard, and then demonstrate it on a giant-screen television that all the girls can see. Most of the class time, however, is allotted to hands-on work at the computers by the girls. This is because the best way to acquire an understanding and appreciation for mathematics is to actively use it as much as possible, and thus we want to give the girls ample opportunity to do this. In keeping with the idea of learning by doing, we use a project-oriented approach in the computer classes. Projects give the girls a chance to tie together all the ideas that they have learned thus far and incorporate them into their work, and thus challenge them considerably.

The projects we use are specifically designed so that cover several important mathematical topics, particularly ones that students often find difficult. The projects emphasize both understanding the ideas, as well as knowing how to use them. Some of the topics we explore with the girls are the perimeter and area of a circle, ratio and proportion, the trigonometric functions, and transformational geometry. In addition to emphasizing the mathematical aspects of the projects, we also teach students about structured programming. Modularizing one's work requires a kind of logic and formalism of its own, just as does mathematics, and thus provides further opportunity for the students to be exposed to this kind of reasoning. In addition, it gives students practice using higher-level thinking skills, which then carries over into their classroom work when they return to school in the fall.

The one requirement of the computer portion of the program is that each girl complete a final project before she leaves. The final project is similar to the other projects that the girls do all summer, with the added feature of giving each girl a chance to explore a particular topic that interests her in depth. Usually this is done starting at the end of the fourth week, when the girls have acquired enough general programming knowledge to be able to implement something on their own. The projects must either explore a mathematical topic or else require the use of mathematics in their implementation. The computer room is open in the afternoon and evenings as well, so the girls are free to come if they want additional time to work on these, or help in doing them.

Logo is an ideal language to use for Summermath program because it lends itself nicely to mathematical projects and activities, and is in keeping with the program's philosophy of learning by doing. During the six weeks of the program the girls actively experience and use mathematics as a tool, something which many of them have not been accustomed to doing before. As they do so they grow to understand the concepts better, as well as to appreciate the importance of what they are learning and to realize the relevance of mathematics in their lives. It is refreshing to see the change that occurs in them during the course of the six weeks. Many start out as passive learners, students who depend on us for initiative and guidance, and who are reluctant to try difficult projects. However, by the time the program draws to an end, they have become active, independent, self-starters who look forward to and very much enjoy a challenge. Perhaps even more important is the fact that the work they do at Summermath, especially the Logo projects, helps to increase their self-confidence considerably. As a result, when they return to their schools in the fall, they have a much more positive self-image because they realize that they too have the ability to tackle and master mathematical material, a feeling that remains with them for life.

What Good is Object-Oriented Programming?

Jeremy A. Jones

Coral Software Corp.
PO Box 307
Cambridge, MA 02142

Object-oriented programming languages are starting to become available as either whole new programming languages like Smalltalk, or as extensions to existing programming languages, like Object Pascal, Objective C, and Neon. Object Logo is a version of Logo that has been given the features of an object-oriented language with the addition of a few new primitives. These new primitives can be used interactively and incrementally in the traditional Logo programming style. This facilitates the introduction and use of the interesting and powerful concepts of object-oriented programming. This paper explains what object-oriented programming means and describes several types of things it is good for.

What is Object-Oriented programming?

Obect-oriented programming means different things to different people, but all interpretations have one thing in common: the programming environment supports a special data type called *objects* that have both state and behavior. A Logo turtle can be thought of as an object whose state is its postion, heading and pen-state and whose behavior is defined by the commands that it will respond to, like FORWARD and RIGHT. However, Logo is not truly object-oriented since users cannot create their own objects.

Objects can be thought of as a collections of variables (state) and procedures (behavior). Each object can have its own versions of particular variables and procedures. For example, a turtle object could have its own version of FORWARD, say, that draws dotted lines.

Objects have a mechanism for sharing their state and behavior called *inheritance*. Once an object is created, it is often useful to be able to create other objects that are similar to it. It would be inefficient and inconvenient to have to duplicate its procedures and variables, so Object Logo allows you to create new objects that are "just like" existing ones. The new objects are said to *inherit* from their prototypes. The new objects are called *descendants* or *children* of the prototype objects, which in turn are called the *ancestors* or *parents* of the new objects. Inherited procedures and variables are not copied, they are shared. Objects can acquire their own versions of procedures and variables, in addition to their inherited ones. For example, you could create a turtle that inherits the moving and drawing behavior (procedures) of the Logo turtle while maintaining its own position and heading (variables).

What is Object-Oriented Programming Good For?

It facilitates the creation of rich computing environments.

Microcomputers are becoming more and more powerful. They now have enough memory to load a large number of Logo programs at one time. It would be nice to have a wide variety of procedures available all the time, but imagine the nightmare of keeping track of them. You would have to keep track of what procedures depended on what, you would have to worry about name conflicts, and you would have to organize things so you could find them. Objects provide a very convienient way to do this. Related procedures and variables can all be kept together in objects so that you can see what they depend on, and procedures can easily be shared using inheritance. Objects can be used to build modular systems where each object has a "published" protocol - the set of procedures and variables accessible outside of the object - that allows it to be used without knowing anything about its internal details. Inheritance also allows objects to be organized in a hierarchy where specialized objects inherit from more general ones.

It encourages exploration and learning by example.

Having a wide variety of organized and accessible programs is great for learning. It is much easier to learn from seeing how things work than from reading explanations. Existing programs can be organized into conceptual units that can be easily "browsed". Such exploration is encouraged by having numerous examples to look at, modify, and experiment with.

It is easy to "get your feet wet".

After looking around this rich environment to see how things work, it is easy to try your hand at modifying it without the fear of permanently altering anything. You can make a new object that inherits some existing state or behavior, and redefine procedures in it without affecting the state or behavior of other objects. Knowing that you can play with things without breaking them encourages exploration. This goes right along with the Logo philosophy of learning by doing.

It facilitates the construction of general purpose programming tools.

Anyone with much experience in programming knows how repetitive it can be. How many times have you written a binary search routine or a specialized sort? A general purpose object-oriented sort can be written that will take any ordered collection of things and sort it. The collection object needs to be able to get and store things by index, and the objects to be sorted need to be able to compare themselves to the other objects in the collection. If you have a list or an array of some objects, you can simply define greater-than for those objects and then sort them automatically. Smalltalk implementors have developed a comprehensive set of such general purpose programming tools for their systems.

Different behaviors can easily be combined.

Object Logo allows you to create objects that inherit from any number of other objects. For example, if you have a voice-generating talker object, you can make a talking turtle by making a new object that inherits from both the turtle and talker objects. Object Logo also provides a simple mechanism for combining the behaviors of several inherited procedures into one. For example, when you define a procedure DRAW, you can then say USUAL.DRAW anywhere in the definition. This means do the *usual* DRAW, or the DRAW that would have been used if the current DRAW procedure did not exist. If all of the inherited DRAW procedures do a USUAL.DRAW, then the new DRAW will perform a combination of all the inherited DRAW procedures.

It simplifies the creation of user customizable systems.

Using objects, a system such as a spreadsheet or word processor can be designed with customization in mind. Each piece of a complex system can be a black box that can either be used as is or customized. This kind of modularization is generally a good practice since complex systems need a lot of refinement. An entire system could be an object that is used as the prototype for a new, user-customized system. A whole new genre of applications can be written in Object Logo that encourage people to figure out how they work and tailor them to their own needs

It is easy to have multiple views of the same object.

While programming, you often want something to appear in different ways. If you have some data in an object, you might want it to appear in tabular form, as a bar chart or as a musical melody. A simple way to do this is for the object to have several children objects, each of which has its own DISPLAY procedure. Since objects have full access to their parents' variables and procedures, they can display that information in their own specialized ways. The view that each of these objects displays can then change as the parent object changes. View objects can be used for any type of transformation, such as data reduction or the correlation of several different parent objects.

It is a easy to network object-oriented systems.

An object-oriented programming environment can itself be thought of and implemented as an object. If many such systems are networked together, they can access each other in the same uniform way that all objects are accessed. They can ask each other to do things, or even inherit capabilities from other systems. People with networked computers can easily share objects with each other, encouraging interaction and group projects.

Proposal for Presentation
LOGO 86

Submitted by:
Anne Bergeron and Renaud Nadeau
Logo Computer Systems, Inc.

Building Microworlds in an Object-Oriented LOGO

One of the best-known "objects" in the world of computers is, without a doubt, the LOGO turtle. It has an internal **state** and the capacity to understand commands that **transform** this state. These characteristics are significant in the success of the turtle geometry microworld. Building programming environments and microworlds seem to have a natural place in a language which supports object manipulation.

A demonstration of two examples of microworlds defined with an extension to the LOGO for the Macintosh will be given.

The first is an environment to create interactive animated cartoons. The second is designed to explore the world of motion-related physical phenomena.

Discussion will focus on object-oriented tools used to develop these microworlds.

GENERALIZED TURTLES:
MORE 'OBJECTS TO THINK WITH'.

Joop Ringelberg

LOGO CENTER NEDERLAND

The question 'is there LOGO after Turtle Graphics?' will be answered in a different way in this presentation: by extending the concept of the graphics turtle to a general moving, writing and reading object, and putting this generalized turtle to work in other fields than TG, we think we can offer the user both a greater sense of control over the computer and narrow the gap between TG and list processing.

An important idea behind TG is that of transfer of knowledge. This is best characterized by Papert's adagium of the turtle as an 'object to think with'. So by slightly extending the turtle's basic repertoire it can be turned into a useful device for many applications. As a matter of fact, we only emphasize what is already there: the turtle's ability to read things from the object it is standing on (such as reading coordinates). Consider for example a textturtle, or cursor as it is usually called. Like a common graphics turtle its has a position, it can move in several directions, it writes things - why not have it read from the screen as well? The same can be said of a disk turtle for example. In short, all Input-Output operations can (in principle) be thought of as turtles moving over various surfaces and having them read and write.

Also, the turtle's existence as an object has been made more explicit. Telling a graphics turtle to move forward amounts to explicitly sending it the message to do so:

```
SAY GTURTLE
    FORWARD 100
```

In the same sense, a turtle on the disk is explicitely told to move to a certain file and to read a list:

```
SAY DTURTLE
    TOFILE "MYFILE
    READLIST
```

However, we expect a result from such a turtle action: so LOGO tells us there is a resulting value:

```
RESULT: [ this is the first list in the file myfile]
```

It is obvious that such an expression can be thought of as a functional expression in the sense that it returns a value that could be passed on to another function or yet another turtle. Thus, the important concept of expressions resulting in a value is introduced in the turtle world in a natural way, even though TG is usually thought of as 'dealing with side-effects'.

We have implemented all Input-Output in our system in this way: graphics turtles, text turtles for textscreen manipulations (for example menu's), music turtles for the manipulation of voices, tiles and sprites objects to design *soft buttons* and icon controlled instant environments, turtles to customize disk and tape I/O, printing on printers and reading from the keyboard, turtles for using paddles and joysticks; all these devices can be controled in the same universal way.

One of the prospects of this project is yet further *extending* the object concept. There is no reason why LOGO users should not be able to create new objects, organize them in classes and define new methods and messages. Turtles are indeed good 'objects to think with', but as Papert has noted, turtles are only the beginning: we should have more useful exciting objects to think and play with.

For further information:

J. Ringelberg
LCN
PO BOX 1408
6501 BK Nijmegen The Netherlands

ABSTRACT

Liddy Nevile,
Barson Research,
335 Johnston Street,
ABBOTSFORD, 3183,
Victoria, Australia.

While there is a reasonable amount of literature about the design of microworlds in Logo emerging, there seems to be little written about what makes one microworld more appealing, more powerful, than another. This aspect of microworlds has aesthetic qualities, and as such is perhaps least explicable in computer science terms. Nevertheless, I believe it is the most important, and that the aesthetics of microworlds should be discussed if only so they become describable.

In creating a number of microworlds to support the learning of such concepts as three-dimensional representations of space, isometric and perspective, tessalations in terms of reflections and transformations, data-bases as intelligent collections of inter-active data, the concept of recursion itself, and so on, we have confronted above all the question of what metaphor will best establish a link between the user and the knowledge in the system?

LEARNING FROM LOGO

When the first Logo papers emerged with dreams of children creating meaningful microworlds for their own purposes, and incidently building powerful learning environments in which they could absorb knowledge by osmosis, the imagination of many was fired.

The reality, fifteen years later, is that teachers (as a group) do not know how to encourage this activity on a long term basis, that the learning does not have obvious enough transfer into the standard school syllabus, that programming skills fall far short of the needs of the programmers, and generally that what was promised is being doubted.

We, in Victoria, have been teaching and using Logo for more than ten years, and yet do not feel any urgency to abandon the task. Instead, we battle harder to broaden the cultural support for using Logo, and argue that even when there is not a lot of hard-data research reports which prove the effectiveness of Logo, there is potential which should and could be realised. This is so, we say, despite the fact that the inherent weaknesses in Logo are well recognised, and that for many, the use of Logo is exhausted, after the 'using it for computer awareness' stage.

For the last year I have worked hard to discover some metaphors which I believe are to a large extent the missing links in many Logo users' knowledge base.

The need for simple metaphoric representations of Logo itself have been responded to more formally in recent years. For instance, Brian Harvey has written his chapter on recursion four times in one book, to facilitate the reader's comprehension of this process (1).

In 1985 a number of papers appeared which offered suggestions about how microworlds could be designed in Logo. These were not to be used for teaching programming techniques as such, but as environments for learning about something in particular, e.g. balance. The knowledge which the user would be expected to bring to the activity might vary, and so might the learning which would take place.

In reaction to what appeared to be a 'soft option' compared with those an environment such as Logo could offer, I undertook some work to produce microworlds for school use which would bridge the gap in what I felt was an appropriate way.

The basic design criteria were roughed out as follows (by myself and Colin Fox who has worked on this project with me):

1. that any microworld should have a low threshold entry point in terms of Logo programming skills, even if the programmes themselves use "advanced" techniques;

2. that any microworld should have a subject content which would benefit from the support of an environment such as Logo could provide;

3. that the use of the microworlds should be two-fold: while offering interesting contexts for using and developing programming skills and

activities, they should also offer stimulating subject-specific environments which could be used independently of programming skills;

4. that while each microworld should be seen as a unit in one sense, a number of increasingly sophisticated sets of procedures should be provided for interaction with the microworld;

5. that while all programmes should consist of easily examined and altered procedures, the procedures which control the system should be easily identified and modified for personal use;

6. that as far as possible, Logo and its associated epistemology of learning developing from a fragmented base of knowledge, should be presented in an exemplary fashion.

The last of these design criteria is of course the most elusive.

How can a topic be represented so that a number of people with differing backgrounds and styles of learning can benefit from the same environment?

Andy di Sessa suggests that in time there will be sufficient knowledge about the learning process and the knowledge which people bring to a learning task, for us to design microworlds which can respond to the knowledge of the learner (ii). What seems to be the problem then, is whether, with this knowledge, the microworld designers will choose accuracy or appropriateness as the inclusion criteria for their microworld.

It seems that the stumbling block for many microworld users, is not the smallness and relevance of the knowledge embedded in the microworld, but the way it is presented, accessed, modified, and thus learned.

A simple microworld example might be one with a set of procedures which merely extract an element from a set and print that element. For instance, given a list of nouns, a procedure called NOUN might print one of the nouns, chosen randomly. Some such sets may be subject specific.

When I was using these procedures with a group of curriculum developers in New Zealand who preferred cricket to computers, their attention was captured when they related their work to cricket. Within a short time, the cricket microworld (which had lost its identity as my set of procedures) contained an extensive data base of information about cricket. There were lists of the acceptable actions which take place on the field, of the positions in which cricketers are found, of the scoring which results from certain events, of illegitimate cricket acts, and even of the comments which Aussie commentators make about New Zealand cricketers.

What were the critical elements of this microworld?

I believe the original starting point, a very simple sentence of the type

DOG BITES BILL

and which became

NOUN VERB NOUN

was the key to the exercise.

In a number of ways this was a metaphor for the collection of cricket information, for a data base. The information about what happens was easily gathered, and recognized as such, under the heading ACTION. Further information stores could be made under the heading NOUN but more headings soon needed to be added. All the time the metaphor was used to support the activity: each time new classifications were developed, sentences using the new information were run to test the material. The relationship of one piece of information to another was testable according to the "does it make a sensible sentence?" test. In the same way, the information itself could be tested according to whether the sentence conveyed a meaningful cricketing situation, or not.

This is not a new, or unusual, use of Logo but the example is often trivialised unnecessarily, I believe.

When computer users are struggling to use computers to work with information using other data schemas, they are often confused by the results they achieve. I believe that moving from paper and pencil collections of data to computer-based ones is often a wasteful exercise because the computer is expected to be just a bigger electronic collection modelled on the paper and pencil version. The power of a collection of data in Logo is of another dimension, and until the user can create a metaphoric representation which can be used to consider what is being done by the computer, the computer's power will be neglected.

Mike Sharple's boxes are a good example of this (iii). He has created a microworld which uses the metaphor of a box as the container into which some information can be put. The problem is, however, that the information which goes into one box may well be a box itself, and other boxes may be expecting access to the information in that element/box too. Thus the model of this data collection is difficult: it is not a hierarchically organised set of information, it is not a set of records which is related by simple matching, and so on. It is not just a computer version of what can be done with paper and pencil. In fact, the model has the same sorts of attributes as the recursion one which Brian Harvey was dealing with in his book. It is, in a sense, merely an extension of the NOUN type example above. But the difference is that the extra dimension which is added is not trivial or easily explained.

I believe that to a large extent the value of a microworld depends upon the appropriateness of the metaphors upon which it is designed. The sentence is so simple a metaphor that it is often not recognised for its role. It is so powerful that its role can be redefined time and again according to a very wide range of situations, styles, and users.

I hope that in time we will be able to learn to look for good metaphors, and so be able to offer some more satisfactory answers to those who question the value of Logo in education.

(i) di Sessa, A., **Knowledge in Pieces**, Address to the Fifteenth Annual Symposium of the Jean Piaget Society, 1985.
(ii) Harvey, B., **Computer Science - Logo Style**, McGraw-Hill, 1985.
(iii) Sharples, M., **Cognition, Computers and Creative Writing**, John Wiley and Sons, Chichester, 1985.

Teaching Pascal With A "Logo" Philosophy

Ursula Wolz
Columbia University

1. Introduction

At first glance it would seem that the teaching of Pascal and the teaching of Logo would require different methodologies and philosophies. Logo encourages an exploratory approach to programming whereas Pascal requires a rather strict top down design discipline. Yet in an introductory freshman programming course at Columbia University, a Logo perspective proved effective in conveying basic programming concepts, and in motivating students. This presentation will describe how a "hard core" Logo teacher taught a Pascal course in a formal classroom setting. It will discuss the curriculum and objectives required by the Computer Science Department - which are an extension those on the Pascal Advanced Placement Exam, and how Logo teaching methods, such as exploration, debugging, personalized projects and an emphasis on "powerful ideas" were used. In particular, the presentation will discuss how recursion which is even "harder" in Pascal than in Logo was introduced.

2. The Course Objectives

Although this course was an entry level computer science course at an major university, the basic content of the course bears strong similarities to the Advanced Placement Exam in Programming. The course is intended to teach potential computer science majors the fundamentals of Pascal and introduce them to good programming practice through topdown design and stepwise refinement. The course is also intended to cover the basics of computer literacy such as the parts of a computer system, and to introduce ideas from computer science such as algorithm analysis.

The course has a well established curriculum which relies on a required text. Individual instructors have some latitude in how they teach the course provided their students become "proficient" in Pascal. The students are extremely passive, since the course has a lecture format. Assignments are usually small programming exercises with known solutions. Furthermore, the compiled nature of Pascal, along with its cumbersome syntax require an attention to detail that often subverts any emphasis on the larger programming ideas.

Under these constraints it became apparent that to teach this course successfully and at the same time maintain some integrity, a modified Logo approach would be necessary. The next section outlines how both the course and a Logo approach were adapted.

3. Using the Logo Philosophy

The fundamental goals, as seen from a "Logo" perspective would provide the students with a sense of how to go about tackling a complex programming problem, how to debug it, and to realize what kinds of limits exist. Furthermore, it was hoped that students would develop some of the "hacker" mentality to think their way through a problem alone or with peers and without resorting to experts. Finally, it was hoped that students would come to understand the "powerful" ideas of programming such as proceduralization in a context beyond the syntactic limits of Pascal.

In order to do this, the course content was organized around the fundamental constructs of programming, rather than around syntactic topics of Pascal. An emphasis was placed on how to implement a particular principle such as iteration or modularization, rather than describing the nuances of the differences between specific Pascal commands.

The class itself relied as little as possible on lecture. The students were expected to have read the appropriate chapters from the text before class. Although some time was spent introducing specific aspects of Pascal, most of the class time was spent in a socratic dialogue between the instructor and students developing solutions to problems that illustrated a particular construct. The major emphasis was on thinking about how to solve problems. When specific topics such as storage needed to be introduced, concrete models such as boxes were used. For example, in order to demonstrate how a program goes from statement to statement, the instructor "stepped through" the instructions on the floor, much as a young child "plays turtle".

Homework assignments were given every other week and were graded. They were designed to separate programming constructs from Pascal syntax, and to give practice in both. Each assignment consisted of three parts, a modify problem, a debugging problem, and a programming problem.

The modify problem consisted of a correct Pascal program which the students were expected run and then modify. It was intended to model techniques for using a particular construct. The modification involved thinking about how another Pascal command could accomplish the same task. For example, one assignment contained a number of procedures. The students were asked to identify which procedures might be better expressed as functions, and to modify the program accordingly.

The debugging problem was intended to help students avoid the major syntactic pitfalls of the Pascal commands that were being introduced. The problem consisted of a carefully constructed buggy program, with a description of what the program was supposed to do. The program also contained semantic bugs that beginners are usually likely to create. The intent was to give the students experience in encountering certain kinds of bugs, and to encourage them to think about how to eliminate them. Furthermore, the semantic bugs forced them to "step through" what the code did, and often shed light on how exactly a particular command worked.

The programming problem was intended to give the students an opportunity to see a problem through from start to finish. It was hoped that the modify and debug problems had provided sufficient practice in Pascal, so that the focus in this part of the assignment could be on the solution design and not on syntactic detail. Whenever possible, the problem was left slightly open ended, so that students could use their creativity and initiative to come up with solutions. Needless to say, there was rarely one "correct" solution.

4. Recursion in Pascal

Recursion had the potential to become a rather difficult subject. It is not easy to express recursive ideas in Pascal. Concrete models such as a recursive turtle procedure that calls itself and causes a turtle to run infinitely do not exist in standard Pascal. Furthermore the text introduced recursion near the end, only in the context of mathematical induction, and without any strong concrete models.

The goal in teaching recursion was to make the students' exposure to recursion gradual and as concrete as possible. Models were first introduced that illustrated recursion as a phenomenon that appears as an allowable expression of Pascal syntax. Later programs were introduced that carefully traced the stages of the recursion, much as a turtle traces the levels of a recursive Logo procedure. Only at the end was recursion put in the context of induction and its place as a powerful expression of mathematical ideas.

5. Summary and Recommendations (If I had to do it again)

There were many constraints on the way this particular programming course had to be taught. Whether the approach is the most effective introductory computer science course is not at issue. Rather, the course was intended to teach good programming style using Pascal. For better or worse, the content of this course is becoming a national norm. I believe that the teaching of such a course can be greatly benefited by a "Logo" perspective. In a future version of this course I would like to incorporate even more of Logo. In particular, I would like to spend the first third to half of the semester using the Logo language, with the same approach in class and in the assignments as described above. At first, the models would center on turtle procedures with a gradual shift to list processing. Only in the second part of the course, when the "powerful ideas" had been adequately presented, would I introduce Pascal. The remainder of the course would be spent exploring the differences and similarities between expressing solutions in Pascal and Logo.

LOGO 86. The Third International Logo Conference. M.I.T., U.S.A., July 9-11, 1986.

TOWARDS AN ARTISANAL USE OF COMPUTERS

Their application to the design and study of three-dimensional forms

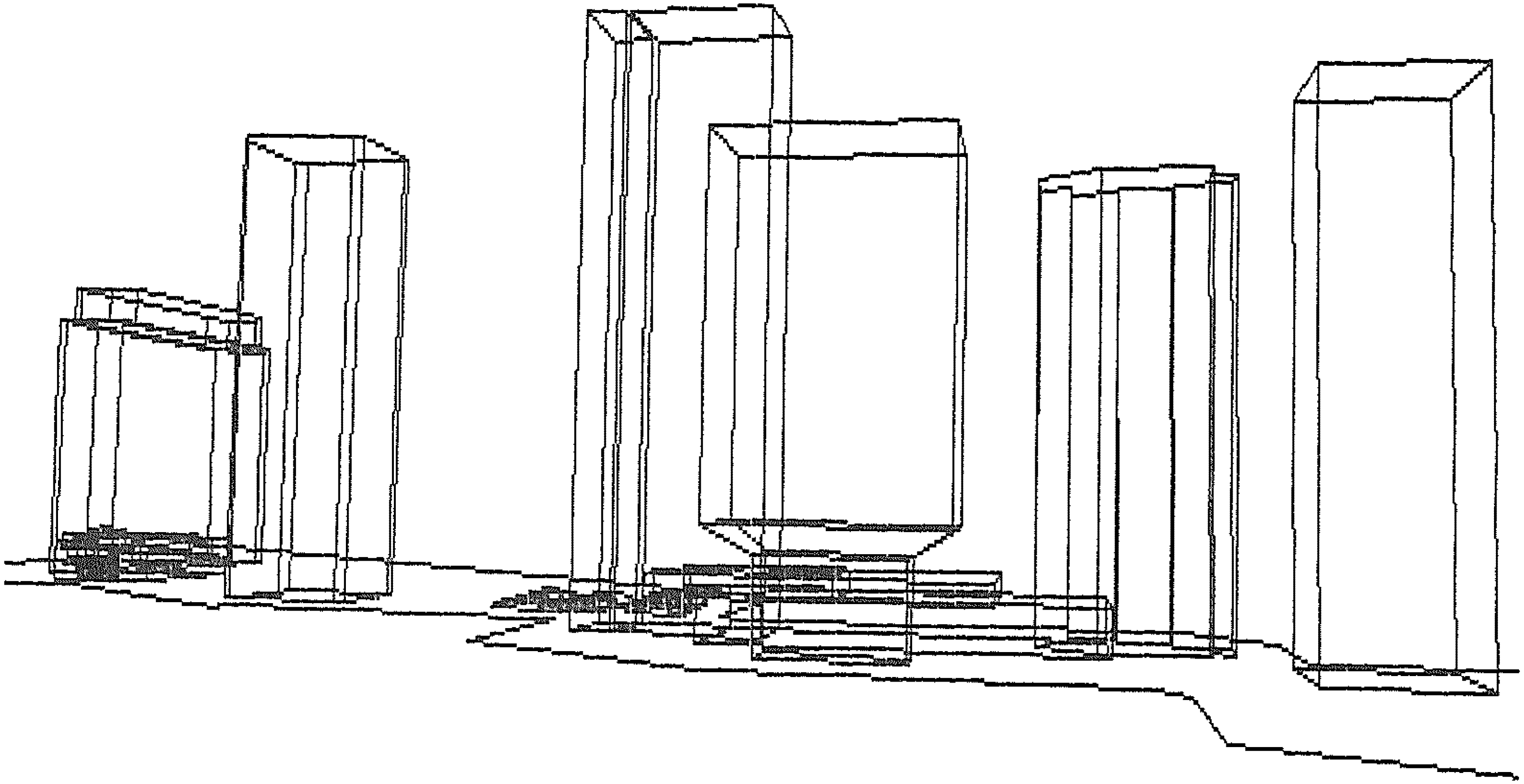
by HORACIO C. REGGINI
Buenos Aires, Argentina

In any discussion of computers in any field, among the most important factors to be considered are the people and cultural settings around computers. These beliefs are keystones of this presentation dealing with the computer as a medium of expression for the artisanal description and creation of three-dimensional forms.

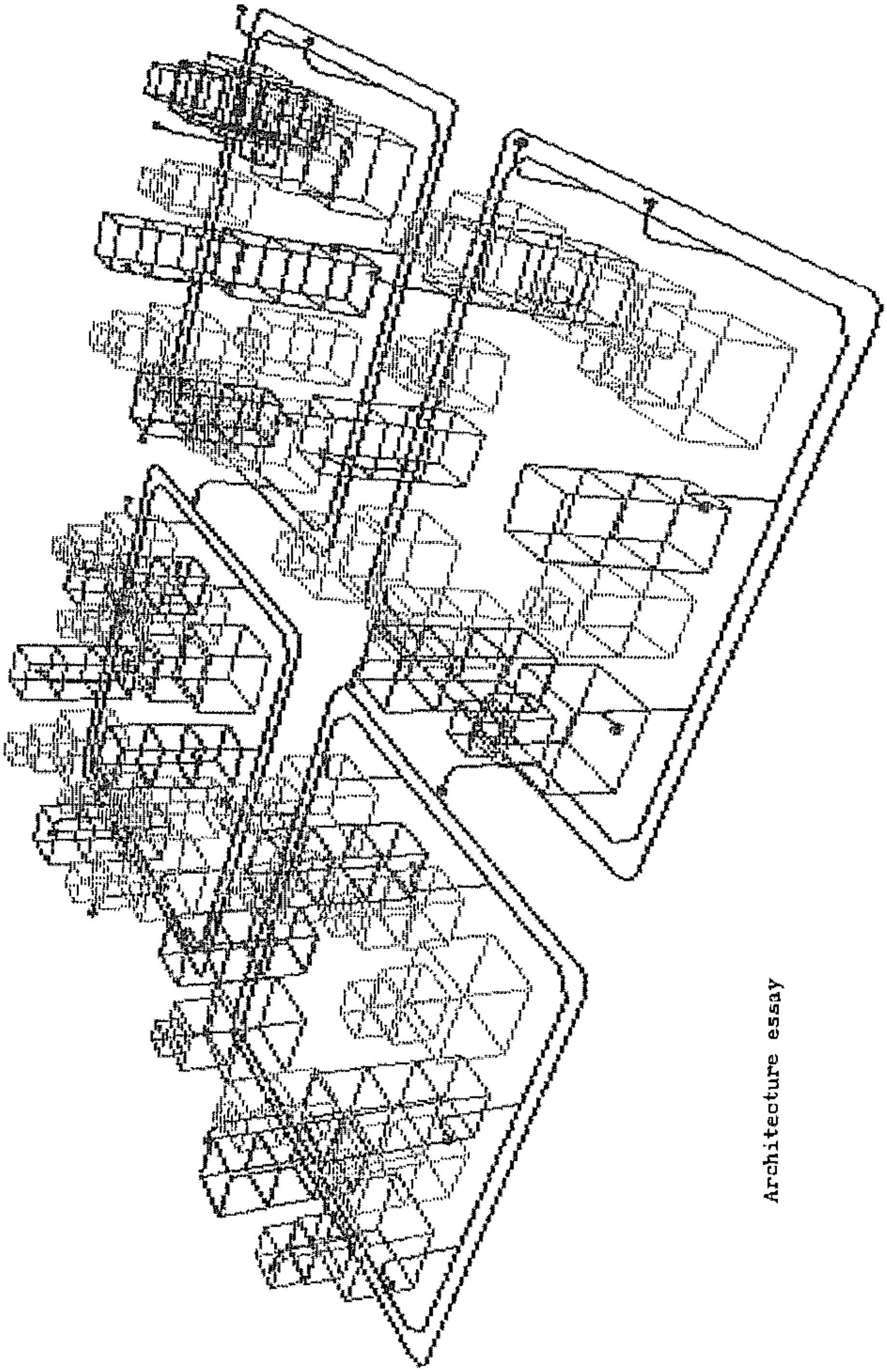
The process a person follows to define a form with Logo* resembles the work of an artisan. Like the artist, the artisan pauses frequently while she works, and in accordance with her progress, she often corrects or changes her plans. The tools the artisan uses are neither complicated nor do they require deep knowledge to use, and every accomplished form carries in itself a characteristical seal that shows the style, knowledge and aesthetics of the artisan who made it. It is my conviction that it is possible and desirable to follow an artisanal approach using a modern instrument like the computer under the control of a simple and powerful language like Logo. The integration with the machine is intense and straightforward; the user works through trial and error, and continuously modifies the work. The procedure that defines a form reveals personal idiosyncrasies and abilities because the same form can be described by different people in different ways. The computer can be used as a versatile medium that only acquires efficiency in the hands of people with the imagination and skill for creative work.

My presentation consists of two parts. The first part shows the application of 3-D-Logo to the design of forms in space, showing once more the power and elegance of intrinsic differential geometry and modular procedural thinking. The second part is devoted to the application of 3-D-Logo to the study of some architectural landmarks like the famous St. Mark's Square in Venice and Michelangelo's Campidoglio in Rome.

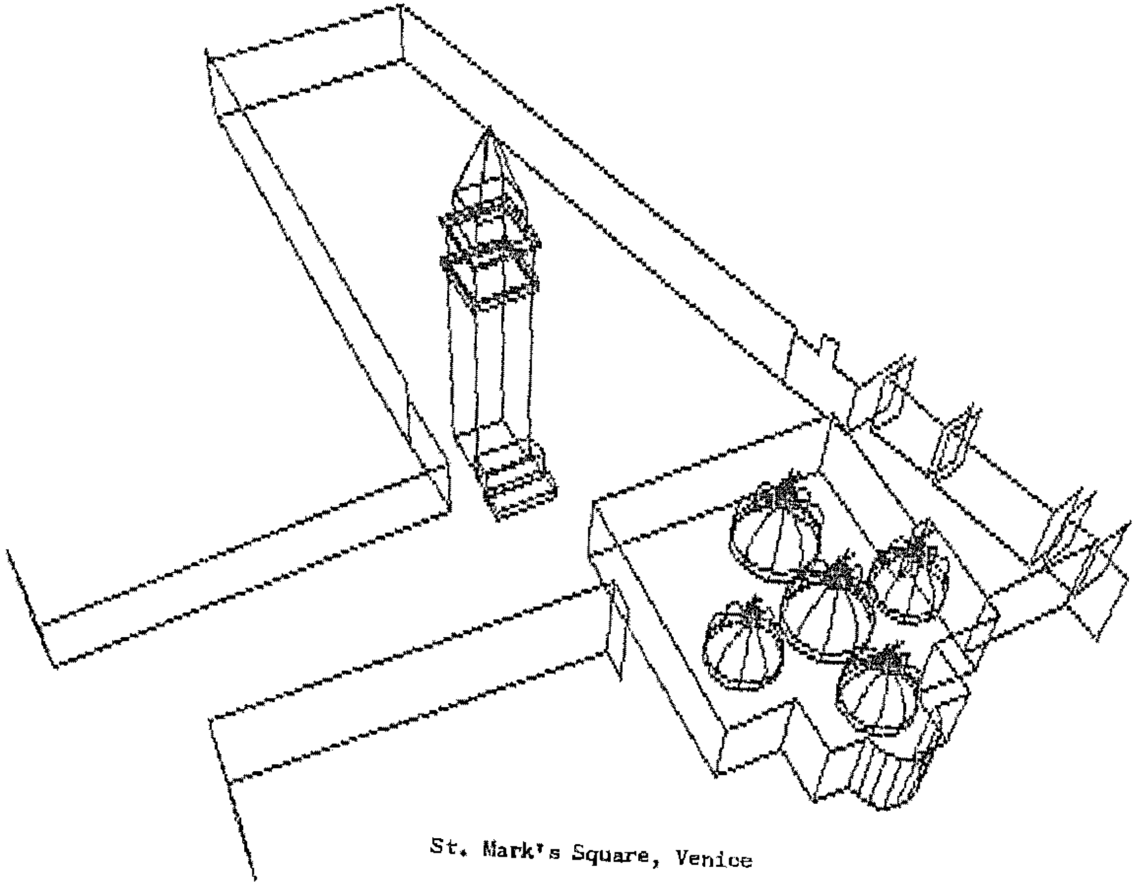
*IDEAS Y FORMAS: Explorando el espacio con Logo, Horacio C. Reggini, Edic. Galápago, Buenos Aires, Argentina, 1985 (IDEAS AND FORMS: Exploring three-dimensional space with Logo).



Group of buildings, Buenos Aires

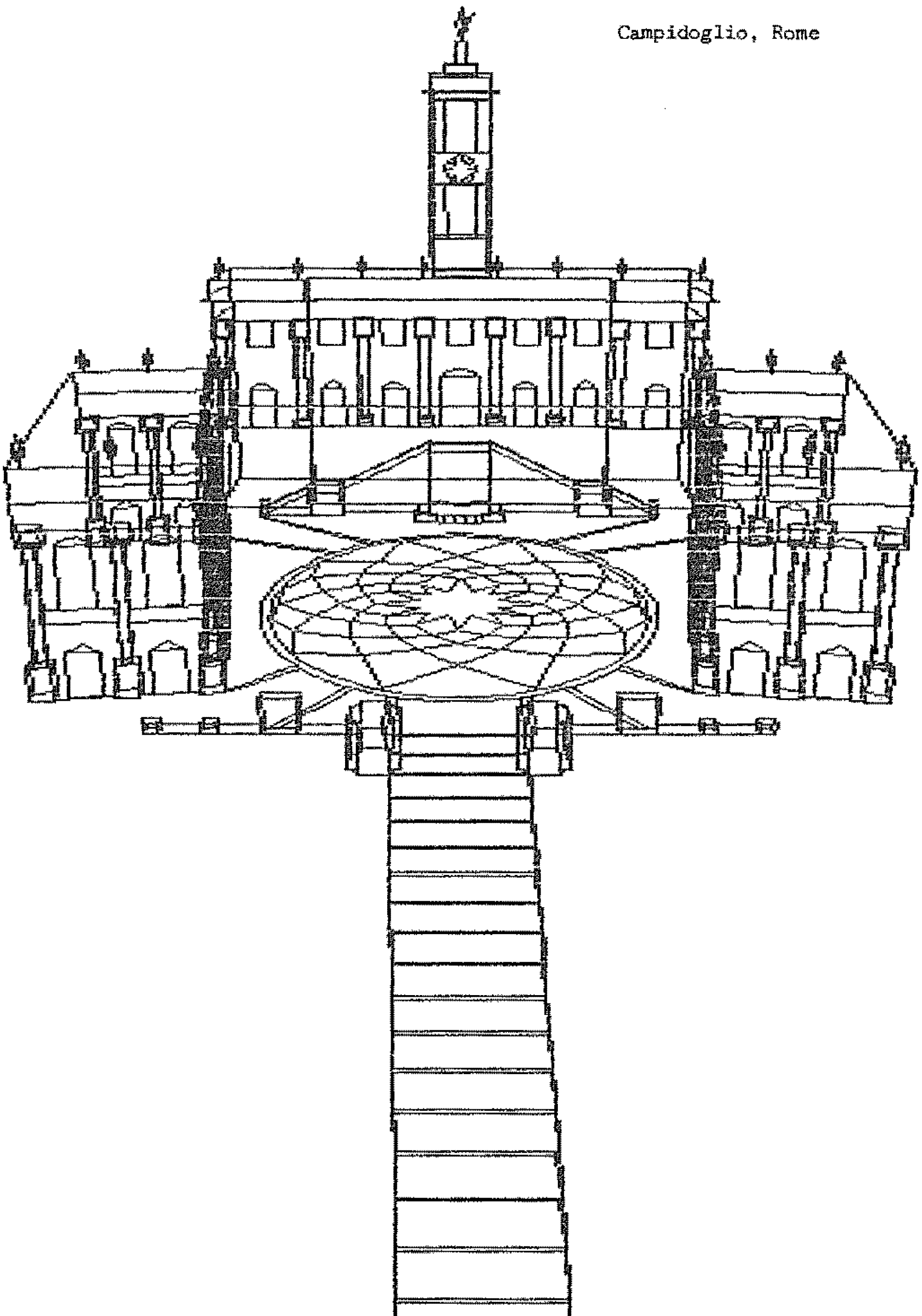


Architecture essay



St. Mark's Square, Venice

Campidoglio, Rome



-- Logo in China --
-- By Molly Watt and Daniel Watt --

Logo In China

by Molly Watt and Daniel Watt
Educational Alternatives
Gregg Lake Road
Antrim, New Hampshire 03440

What are the possiblitities for Logo in China? Do Seymour Papert's concepts of "Piagetian learning, ... learning without curriculum," and "the computer as pencil," readily available to all learners whenever needed or desired, have meaning in a highly centralized nation of 1 billion people? What would educators brought up in a system that emphasises large group rote learning, make of Logo's emphasis on learning by doing, and children as builders of their own intellectual structures?

As we write this, we are preparing to spend four weeks teaching Logo to some of China's leading educators. During the month of June 1986 we will conduct an intensive hands-on workshop for 40 educators in Beijing, at the invitation of the Curriculum and Teaching Materials Research Institute, the organization responsible for curriculum development for hundreds of millions of elementary and secondary school students throughout China. Our plan is to present an overview of Logo programming and educational activities, with particular emphasis on the powerful ideas embedded in the language and on learning through hands-on experience. In addition to sharing our collection of Logo microworlds and activities built by teachers in the US and Canada to connect Logo to a number of different curriculum areas, we will work closely with workshop participants to create as many connections as possible between Logo, and China's educational needs and aspirations.

In our presentation we will share our experiences in China, with examples of Logo projects carried out by workshop participants, and slides of the workshop setting and activities. We will describe the reactions of the workshop particpants to Logo, and discuss our sense of the implications of Logo's philosophy of education for education in China.

LOGO AND INDIVIDUALITY:
Personnal patterns of learning in Logo
Models to promote individuality: Logo-video-clips

by Francine Bonnier
Pierre Tremblay

If Uri Leron raises the question of "two LOGOS" refering to two kinds of LOGO language, we raise the question of "two LOGOS" refering to two kinds of LOGO practice:

1) A Logo that praises **conformity**: Logo being taugh to students step by step, or being used to teach a known concept, in a known way. Exercises and lessons are then used in a formal traditionnal way, organised in a curriculum with predetermined objectives. In this practice, children are juged unabled of learning by themselves in an efficient profitable way. The teacher is a teaching adult controlling what is being learned.
2) A Logo that praises **individuality**: Logo is given to children so they can work on their own, building **their own patterns of learning,** structuring their own way of thinking and being able of creating their own objectives. The learner is the person controlling what is being learned. The adult must not teach, but be a resource person, a facilitator.

We have been working with this later kind of LOGO in two school settings, one for deaf children and one for regular children. We have looked in the recent LOGO litterature and have found very few people working in this same perspective. Is this why we have then heard so little about new paths in learning, so few about changes in patterns of learning or changes in the social structure of learning?

In our daily practice of an "untamed LOGO" with children aged 5 to 13, we have observed and gathered information on a great variety of personnal patterns of learning built by the children to structure their own way of thinking:

-Patterns modeling the choice of digits.

-Patterns modeling the actions the child decides to undertake: patterns to thame the keyboard, the turtle, a primitive; patterns to explore, to grope about, to try, to aimed at an objective; patterns to search for new ideas...

-Patterns modeling a reaction to a LOGO message: patterns to answer, patterns for debugging, patterns for trying again, patterns for checking over...

-Patterns modeling the keeping of a piece of work: patterns for copiing, taking notes, putting into procedures, saving, printing, using files...
-patterns modeling the organisation of one's work, patterns for starting, ending a task, patterns for collaborating with another person, patterns for seeking or giving help...

In the first part of our presentation, we shall illustrate a few of these individually designed patterns by giving examples taken directly from the children's work. What seems so astounding about these patterns is their personnal touch. Even if two children have very similar patterns, there is always a difference that is very important to each one of them. Like a pair of shoes, the more you use them, the more you make them yours: the more you are at ease with them, the more you use them in different situations or for different purposes and the more you can accomplish and learn with them.

In the second part of our presentation, we will use Logo-video-clips as models to show how individuality, originality and creativity can be promoted in a Logo culture. As an example, one video-clip is titled: "A turtle that expresses herself". In this short video one can see a dancing turtle, a turtle-torpedo sinking a ship, a turtle crashing into a wall, a turtle falling down a stairway, a turtle disappearing into a hole, and a turtle telling a story.

Finally, we will discuss with the participants the question of two Logos in sharp contrast. Does this mean two very different trends in education? Two different philosophy? Two different kinds of attitudes towards the learner? Is a neat choice to be made (one not being able to choose a midline, a melting pot of both being impossible)?

Ilse Schenk
Feb.25, 1985

LOGO 86 PRESENTATION

Title: IN TOUCH -
with LOGO

With 20 people saying hello to each other,- how many handshakes are there?

At the Fenn School in Concord, a class of 5th graders became quite excited when guesses and answers to this question led to an interesting math problem, and some very graphic problem-solving with LOGO.

LOGO-pattern-thinking became very real and a part of life they could TOUCH, that made SENSE.

Will my presentation of slides, videotapes and LOGO animations meet the challenge of capturing and recreating the lively athmosphere of puzzlement and wonder, excitement and discovery, which happened in this classroom and others, where LOGO-adventurers of all ages get IN TOUCH with a new way of seeing?

I shall certainly do my best to make it so -

Coming to LOGO from the visual arts, my approach has always been and remained a primarily visual, sensory and intuitive one. Quite consciously and intentionally, I never went far beyond the "beginner's stage" in Turtle Graphics, both in my own work as well as in my teaching.

One of the reasons for this is my fascination with the freshness and spontaneity of the BEGINNINGS, the discovery stage of anything. A new day, a new year, a new life, a new experience -

Isn't it most often at the foundation, the fundamental level of anything, that one is closest to the great simplicity of essence?

Another reason: Right from the start, before knowing much more than FD and BK, RT and LT, WRAP and REPEAT, and of course PENREVERSE,- wonderful and often almost magical opportunities suggested themselves to me for using LOGO graphics and animation as tools for visual thinking and symbol-making. Having since gone only some small distance beyond these fundamental beginnings, I have never yet come even close to exhausting the creative potential of these powerful tools.

In comparing notes with my LOGO collegues, it seems that I have created, without really intending to, my very own LOGO "Microworld", in which the first and foremost principle is to keep IN TOUCH with my own learning experience; to become intensely aware and get IN TOUCH with the fundamental realities of the physical, sensory world around me, by seeing them reflected in the motion and rhythms of my LOGO patterns.

Something very similar seems to be happening to my students,- children and adults alike.

From setting in motion and reflecting on some basic patterns and LOGO rhythms, there is a logical step to getting IN TOUCH with the fundamental quality of rhythm and intermittency of natural phenomena, with one's breath or pulse-beat the most immediate at hand.

And what perhaps intrigues me most essentially:

Regardless of age or interest group,- I find that in working with people at this level of fundamental discovery and archetypal symbology,- a quite genuine excitement and environment of personal growth develops spontaneously, putting participants IN TOUCH with themselves and each other.

It is my hope that something of that very nature will happen as I'll share this presentation with you.

VICKI CARVER. COORDINATOR, PLAYING TO WIN COMMUNITY COMPUTER CENTER
1761 Third Avenue (Rear Entrance) NY, NY 10029 (212)369 4077

In the spring of 1981 I became committed to learning Logo, to learning with Logo, and to becoming involved in the work of the Logo group -- for three reasons.

The first was to learn math. I was a math-dropout. At age 32 I read ZEN AND THE ART OF MOTORCYCLE MAINTENANCE and, for the first time, had an inkling that there was a whole world of creative, controversial, intriguing and relevant mathematics and that it seemed to have been systematically hidden away from this basic, working class, public school kid. I read GODEL, ESCHER, BACH, by Hofstadter, and discovered that this hidden math was incredibly funny, but not easy. Then I accidentally stumbled upon MINDSTORMS and knew I had found a road I could follow to this math without getting too lost too often -- a road with familiar markings.

The second draw was political. I wanted to learn about computers, because it seemed obvious that technology was the critical element in the next socio-economic revolution. Computers seemed to be a fundamentally new kind of tool with a powerful lever effect. Information and expertise in this area was empowering, and I was uncomfortable with a state of affairs in which a fairly narrowly defined group of white men held most of that information and expertise. I wanted some, and I felt the world I lived in would be safer and of better quality if people of all varieties shared in the development, care, and use of this new technology. But, how do you get the information and expertise in the first place? It is inevitably filtered through those who currently hold it. The first teachers come from the group "in power". Again, my reading of MINDSTORMS told me this expert and teacher was aware and willing to speak on the politics of choice in machine design, language design, teaching practices, public access, and so on. Papert continues to be among the very few in the field who regularly speaks on these issues with students, teachers, and the public at large.

The third attraction grows out of the other two. Because of my own life experience as a student (a female, first generation college graduate from a working class Midwestern family), I choose to put my energy toward the politicizing and empowering of people in oppressed and excluded groups. The particular focus of my work, again based on personal issues, is helping people recognize their inherent intelligence and the value and extent of their learning through life experience, encouraging them to apply that intelligence to solving personal and societal problems, and enabling them to use the intellectual acumen they've developed so far to take charge of their own learning and extend their study to any area they choose.

Logo is a wonderful tool for use in such work. It provides a medium for expression of skills and intuitions that have been developed in less academic pursuits. Because this expression takes place in a high-prestige context (The Computer) and bears obvious connection to "higher" academic arenas (mathematics, physics, and linguistics), it makes a powerful impact on the process of reversing institutionally enforced and generally accepted stereotypes of whole groups of people --stereotypes that are often bought into by the struggling student.

I remember, in my mid-30's, discovering that there was such a thing as the heuristics of problem-solving and that I was good at using them and that I had developed these skills over 15 years of tap and ballet dancing and as many more years of figure drawing. Drawing is seeing, with the eye and the hand-arm-body, and seeing is framing and reframing and reframing as many different ways and from as many different viewpoints as it takes to see the thing well enough to draw it. It is putting oneself in the place of the model in order to feel the muscles that are contracted and stretched and where there is tension and where there is complete relaxation. If you are curious about the heuristics of dancing, come to this session. I suspect we could add basketball and football, auto mechanics, folk medicine, and so on to this list.

In this session, I will address the issues described above from the viewpoint of student and from my experiences as a teacher in community education in a Black Midwestern community and in New York City's El Barrio (also known as East Harlem) and similar neighborhoods.

I believe that any success I have experienced in accomplishing the work that drew me to Logo is owing to the fact that I have examined and reclaimed my own learning experiences and become aware of myself and others as we exist in and are influenced by political and social institutions and theories. This has enabled me to pay better attention to the people with whom I work; to be more creative, flexible, and responsive in the classroom; to use such tools as Logo more effectively; to foster an environment that inspires that revolution in education many of us have sought.

Therefore, much of this session will be spent on participants being invited to tell their own experiences as learners --how did they get so smart? What do they wish would have been different? How did they learn to solve problems? What is still hard about it? What important thinking skills and expertise have they developed in out-of-school learning? How have oppression issues effected their learning? What's tough about working with students? How are their students effected by oppression issues? What are their students' immediate and pressing problems and concerns? Has anyone tried addressing these issues in class--putting the class energy toward solving the problem? Can Logo analogies be helpful in such discussions? How about the reverse? Do they "think out loud" when they are solving a problem with the class or with individual students? Can critical pedagogy by applied to public school classrooms and how? What are some successes they have seen with students using Logo and discovering they could apply skills learned outside of school to Logo pursuits? How about students applying skills learned with Logo to out-of-school pursuits?

The fact that these questions and issues are inspired by the call for papers to an educational computing conference adds evidence to one of the great surprises to emerge from observations of the increasing educational and personal use of computers by the broad population: That the prevailing effect of computers on people seems to be primarily affective.

Prof. Ruth E. Sower
Cabrini College
Radnor, PA 19087

NATIVE AMERICAN PUPILS' RESPONSES TO A LOGO LEARNING ENVIRONMENT

The focus of the proposed presentation for LOGO 86 will be a project conducted at the Tuba City Boarding School in Tuba City, Arizona. The purpose of the project was to investigate how Navajo and Hopi children learn when using LOGO. An hypothesis of the study was that increased language and cognitive activities in the LOGO programming sessions would lead to improvements in other subject areas that are dependent on language development and reasoning. While the study was conducted for one semester and it is well known that definitive work in this area will require longitudinal studies, it was felt that the present effort might produce some modest changes in some cognitive measures. Also ethnographic data were collected in order to evaluate the relative conditions in regular classrooms and in the LOGO classroom. Student behavior in both settings as well as student talk and teacher talk were observed and recorded for further analysis.

The presentation will include information concerning the following parts of the study:

1. An explanation regarding the unique learning needs of the Native American pupil
2. A discussion of the suitability of LOGO as a learning medium for Native American pupils.
3. Discussion of the project, the subjects, and the method
4. Presentation of the outcomes of the pre/post testing
5. Discussion of ethnographic data collection techniques and outcomes
6. Suggestions for future research based on the present study

The study was conducted from 1-5-86 to 4-25-86 at a Bureau of Indian Affairs boarding school housing 900 primary age students, grades K-8. A randomly assigned group of 30, 4th grade pupils were assigned to an experimental group and 30 others to a control group. All were tested using the Raven Progressive Matrices, the Peaboby Picture Vocabulary Test and the school's standardized achievement tests. Other information collected for evaluation purposes included:number of siblings, whether parents were employed, age,and whether the child lived at the school or commuted.

The experimental group met in groups of six for LOGO instruction for 45 minutes per day, four times per week for a period of ten weeks. In addition to working with LOGO, children were encouraged to use the word processor and write stories to accompany LOGO drawings. Children kept journals of daily activities. They also prepared a book of their work for the entire ten week period. One researcher made detailed recordings of LOGO classroom talk and behavior each day. Regular observations of the experimental group subjects were also made in the regular classroom settings.

The research was conducted as a sabbatical research project by the presentor, R. Sower, Ph.D. Associate Professor of Education, Cabrini College, Radnor , Pa. Some support was provided by a grant from AACTE and some support was provided by the Tuba City Boarding School. A Cabrini College student teacher, Lisa Nolan participated in the teaching of the LOGO classes and in the data collection procedures.

Romancing the Turtle:
Logo in a Multicultural Environment

Working with Logo is like reading a romance novel. There is a sense of exhilaration, creativity, achievement, even the supernatural (the Ah Ha!) come from Logo useage. Logo can be more than a culture, it becomes a passion. When I went to the dictionary, I found the following:

> ROMANCE: a narrative depicting heroic or marvelous achievements, colorful scenes, chivalrous devotion, unusual or even supernatural experiences or other matters of a kind that appeal to the imagination.

The above use of the definition "romance" reminded me of some extraordinary microworlds constructed by some of the children I have worked with. Teaching experience in Oregon, Guatemala, and New Mexico has given me the opportunity to work with various groups of learners from a variety of age, ethnic and cultural groups who have all been romanced by Logo. I have seen these learners resolve many classroom related problems successfully, often in truly distinctive, creative and unusual ways. Given the similar learning environments, teaching tasks, and curricular constraints, it is my opinion that answers appear to be resolved in a universal or "international" manner. For this reason Logo

Daniel C. Orey, M.A.
Department of Curriculum and Instruction
in Multicultural Teacher Education
College of Education
The University of New Mexico
Albuquerque, NM 87131

is a good tool in which to attempt a cross-cultural study between distinct groups of learners.

Cognitive psychologist Robert Sternberg describes intelligence as being constructed of three important areas. The first area is the basic thought or mental processes, which he calls components. The second area discusses the synthetic abilities we have. This is the ability to cope creatively with new situations and practical skills. The third area, the contextual aspect, deals with tacit or implied knowledge that enables people to succeed in everyday life. Intelligent behavior as defined by Sternberg is:

> mental activity directed toward purposive adaption to, and selection, and shaping of, real-world environments relevant to one´s life.

Given certain kinds of experiences, it would seem possible then to develop specific activities for any person. His work may give us a stronger reason to use Logo for the development of certain mental processes. Logo activities and microworlds could therefore be developed that have an appropriate context for the learner. Activities that play upon the romance or passion in the child.

It may be found that the learning achieved by children using the same Sprite activities in one location may be qualitatively different than those done by children of the same age in another place or culture. Logo may be able to assist children from different learning traditions in the acquisition of experiences that Seymour Papert has said help strengthen scientific weakness that exists in some cultures. At the same time it can give scientifically "macho" cultures the ability to experience non-traditional forms of learning.

In the summer of 1983, I traveled to Guatemala with a microcomputer and Logo software. In so doing, I believe this became the first introduction of Logo programming language and its powerful ideas to Highland Maya children. This informal study proved a hunch of mine, as well as demonstrating the feasibility of cross-cultural Logo research that later became a pilot study.

In progress we have a study that is looking at the differences that exist in the mental processing of Logo activities by fourth grade students in three culturally distinct locations in Guatemala, Mexico and New Mexico. I hope to conclude this presentation by discussing the preliminary findings from our pilot study in Puebla, Mexico.

LOGO CULTURES - HUMAN-CENTRED LEARNING SYSTEMS

Karamjit S Gill
SEAKE Centre, Brighton Polytechnic, England

KEYWORDS: Design; Special Education; Adult Literacy; Logo Philosophy; Human-centred Systems; Socio-cultural Context

This presentation will discuss some of the issues on 'Logo revolution' raised by Brian Harvey in his letter on LOGO 86 and will attempt to emphasise the role of Logo as a culture rather than a programming language. It will further discuss how the SEAKE Centre's work has built upon and extended the Logo philosophy for designing human-centred systems for special education, adult literacy and training.

Observing discussion on Logo at various forums and looking at some publications and literature, it is not difficult to form an impression that " Logo revolution" may have changed its course. Logo is presented as yet another(of course for creative learning) programming language suitable for designing and implementing school curriculum. The vocabulary used for presenting Logo e.g. computer graphics, recursion and problem solving is not very different from the vocabulary used for presenting any another programming language within the context of school based learning environments. If we consider the area of computer literacy, Logo is again presented as a programming language suitable for curriculum implementation by many practitioners. The debate on Logo vs Prolog or Logo vs BASIC again gets confused because of its focus on programming rather than on learning philosophies.

It is this presentation of Logo as a programming language that seems to have led many computing practitioners and technically-oriented teachers to " integrate Logo with a certain body of traditional curriculum". This focus on integration has also been influenced by the so called ' information technology revolution' which has led to the development of computer literacy curricula mainly in terms of the computer and computer programming. Slogans such as ' hands-on-experience', key-board skills and word-processing have been propagated by powerful 'interests'groups such as government bodies, training agencies and computer manufacturers with an aim of selling computer literacy to general public as a panacea for all economic and industrial ills of Western societies. Within this scenario of computer literacy, the computer is perceived as a cost-effective tool for traditional education on the one hand, and as a source for providing low-level technical training to new generation of industrial workers on the other hand. It is therefore not surprising that many traditional and technically-oriented teachers and practitioners have tended to use Logo as another programming language for implementing curricula for computer literacy. This technical focus of computer literacy has given a false sense of security to a large number of people attracted to and involved in this literacy compaign. But it has done very little to enhance human skills and human cognitive competences of people for coping with the new technological changes. If Logo is also perceived and seen by public as being used a technical tool for propagating this retrograde concept of literacy, then it is justifiable to be concerned about Logo being ' stripped away ' of its philosophy of learning.

However, if we consider some of the presentations at the LOGO 85 Conference e.g.: MIT HIGH DENSITY SCHOOL PROJECTS [Denise Basaillon], Computer Cultures vs computer Classrooms[Aaron Falbel], Computers and Illiterate Women[Fatima Seye Sylla], On Being Creative [E Paul Goldenberg], Logo and The Arts[Pamela Sharp], then we can take the comfort that Logo philosophy and its culture is alive and well. But these presentations of Logo are centred around a few model projects (which are obviously central to the philosophy) and a small number of committed Logo workers. The focus of these presentations is not on programming but on the learning domains and learning issues such as social interaction, moral and cultural experiences, guided discovery and skills transfer. The difference between this socio-cultural focus of presenting Logo and the 'programming' focus discussed earlier shows a distancing between the " Seymour Papert's philosophy of learning " and the ' programming practice' of Logo. This distancing is further enforced by the predominant emphasis on machine-centred approaches to education and training, which regard human creativity and learning as secondary to acquisition and depositing of technical skills.

This distancing between Logo philosophy and its 'public' practice may go some way to answering Brian Harvey's question: " Whatever happened to the (Logo) revolution?". It is not over yet. It got misunderstood, misrepresented and misinterpreted in many circles. There are, however, positive sign on the horizon: computer literacy hopefully has had its haydays and its purely technical format is likely to fade away; MIT's projects on Logo/Lego should provide pragmatic models of social interaction; Logo culture is becoming extended by the availability of new technologies such as video disc and speech; more and more arts and social science teachers and researchers are bocoming involved in the creative applications of new technology.

We at the SEAKE Centre has always regarded Logo philosophy as a source of inspiration rather than Logo as primarily a programming language. In 1981, we initiated a special education project CAAAT (Computer Aided Animated Arts Theatre) which aimed to develop interactive learning tools for children with learning difficulties. In the initial stages of our work, we used Logo for our prototyping work on language development and soon realised the potential of Logo philosphy for designing learning environments for social interaction. Now this work includes projects on: i) Language and cognitive development for school age children and children with learning difficulties(moderate); ii) Social and moral skills for youth with learning difficulties (moderate); iii) Multi-lingual language development; and iv) Life skills (safety) for adults with mental retardation. All these projects are based on the social communication and life skills needs of learners, and take into account their social and cultural experiences. The design of these projects use programming languages and combine technologies of interactive video (both video tape and video disc), computer animation, text and sound in ways appropriate to the domain of learning and the needs of the learner.

In 1983, we initiated an adult literacy project on ' Basic Education in New Technology and Literacy ' for disadvantaged adults including women from ehnic communities. Logo philosophy of social interaction once again provided a framework for designing an interactive learning system for English as a language for social communication rather than English as a second language in the traditional sense. Women students were actively involved in the selection of the learning domain, formulation of the language learning needs, and the evaluation of the system at all stages of its development. The domain of diet and health provided a knowldge base of the students' diet and health needs and their needs for gaining access to health facilities and resources. The domain also provided knowledge about the health care resources provided by the health authorities, and social communication and training needs of health staff. A prototype program has been developed in Microprolog and will form the basis of an interactive system for diet planning and health. Initially this sytem will be developed for training of health visitors and nursing staff within a multi-cultural context. Later on, it will be extended to meet the learning needs of women from ethnic communities.

From our experiences at the SEAKE Centre, we conclude that:

-Logo philosophy of children-centred learning can be used as a powerful focus for designing human-centred systems for the education and training of disadvantaged adults. However, the domains for adults should focus on their social and life skills needs and build upon their social and cultural skills and experiences.

- Problem solving cannot be divorced from the life experiences and life expectations of the learner. Learning takes place in dynamic social and cultural domains and not in carefully structured environments so beloved by experimental scientists. Problem solving, in the context of social interaction, is about transferring human skills and human knowledge between and across social domains, and is therefore not " amenable to engineering-style solutions".

APPLICATIONS OF TURTLE GEOMETRY, GRADE 12 CALCULUS STUDENTS

Beverly Mugrage
The University of Akron

KEYWORDS: Logo, Turtle Geometry, High School, Advanced Topics, Evaluation

Central- Hower is an inner city magnet school adjacent to the campus of the University of Akron. In December, 1985 both schools began to work together on a project involving two classes of calculus students at the high school. One of the classes of students worked daily on Logo during their lunch hour. On two days a week, an instructor from the university was present to pose challenging problems and to trouble shoot.

Students worked on projects from each of the following areas, their creativity often activated by ideas from Abelson and di Sessa's Turtle Geometry.

1. Models of animal behavior,

2. Vector analysis,

3. Turtle motion in three dimensions, and

4. Topology.

Both classes were tested prior to the beginning of the project on figure classification. The test presented the student with four figures with a particular property, four figures lacking in this property and then asked the students to choose from among five figures that one which belonged with the first set of figures. A number series test was also administered at this time to both groups. Since both classes will study a module in Pascal on which they will be tested at the end of the school year, these test results will also be compared.

Videotapes of student work and of student comments and reactions to their work are being made.

For further information, contact

Beverly Mugrage
Division of Associate Studies
The University of Akron
Akron, Ohio 44325

Mathematically Rich Explorations in Turtle Geometry for Secondary and Undergraduate Mathematics Students

Turtle geometry in the Logo environment provides an excellent opportunity for graphically exploring mathematical ideas. The exploration of poly procedures, in particular, provides an opportunity for integrating Logo, mathematical concepts, and mathematical thinking. Exploring poly procedures leads to the investigation of mathematical concepts underlying these procedures. Mathematical thinking is the vehicle used and developed in this process.

While the use of poly procedures for developing mathematical ideas at the elementary level has been well documented, polys also provide an excellent environment for exploring higher level mathematical ideas with secondary school and undergraduate mathematics students. This paper will describe how two poly procedures and a duopoly procedure can be used with such students. The exploration of these procedures involves mathematical concepts from turtle geometry, number theory, algebra, and vectors. A wide range of problem-solving techniques are used including observing, collecting and analyzing data, writing formulas, inductive reasoning, deductive reasoning, estimating, deriving formulas, solving equations, looking for patterns, looking for relationships, looking for restrictions, and making inferences. The benefits of implementing these explorations with secondary and undergraduate mathematics students, and teacher training groups, will be discussed.

EXPLORATION # 1. POLYS USING REPETITION

```
TO POLY1 :N :R :S
REPEAT :N [ FORWARD :S RIGHT :R * 360 / :N ]
END
```

INPUTS –		
:N	=	number of sides of polygon
:R	=	number of rotations or total turtle trips
:S	=	length of the side

Since this procedure produces both simple polys and star polys, the main purpose of this exploration was to determine what values the variables must have in order for each class of poly to be drawn. The following questions were proposed to be used as guidelines:

1. What restrictions must be placed on the number of vertices (:N) and the number of total turtle trips (:R) for a simply poly or a star poly to be drawn?
2. What relationships must exist between the number of vertices and the number of turtle trips to generate a simple poly? to generate a star poly?
3. How many different :N-pointed polys are possible?

   How many different :N-pointed star polys are possible?
4. What "family of polys" is generated for each :N ?

Concepts used include: total turning, relatively prime numbers, modular systems, symmetry, Closed Path Theorem, and Simple Closed Path Theorem.

EXPLORATION # 2. POLYS USING RECURSION

```
TO POLY2 :S  :A
FORWARD :S  RIGHT  :A
IF HEADING = 0  STOP
POLY2 :S  :A
END
```

```
INPUTS –  :S      =  length of the side
          :A      =  turning angle
```

The POLY2 procedure differs from the POLY1 procedure in that the turning angle is given but the number of sides or vertices of the polygon is unknown. With this shift in emphasis, the main questions to be considered are:

1. Given the turning angle :A, what is the number of sides or vertices in the polygon generated by POLY2?
2. Given the turning angle :A, will a simple poly or a star poly be generated?

Concepts used include: total turning, turning angle, least comon multiple, greatest common factor, properties of equalities, equations, Poly Closing Theorem.

EXPLORATION # 3. DUOPOLY – THE INTERLEAVING OF TWO POLYS

```
TO DUOPOLY  :S1  :A1  :S2  :A2  :C
IF :C = LCM (NSIDES :A1) (NSIDES :A2) STOP
VECTOR :C * :A1  :S1
VECTOR :C * :A2  :S2
DUOPOLY  :S1  :A1  :S2  :A2  :C + 1
END
```

INPUTS – :S1 = length of the side of first poly
:A1 = tuming angle of first poly
:S2 = length of the side of second poly
:A2 = turning angle of second poly
:C = counter

The principle difference between the previous poly procedures and the duopoly procedure is the way in which the polygons are generated. In the duopoly procedure polygons are generated using the concept of vectors. DUOPOLY uses subprocedures NSIDES which outputs the number of sides of the polygon and VECTOR :DIRECTION :LENGTH which draws a vector with specified direction and length. This procedures was explored to answer the following:

1. How is a poly generated using vectors?
2. How is a duopoly generated?
3. When will a duopoly close?
4. What inputs can be used to produce various duopolys?
5. Can three polys be interleaved? four? N? Write a procedure to generate multipolys.
6. What figures can be generated using inputs for a star poly and a simple poly?
7. What inputs can be used to produce "starring" effects?

Concepts used include: vectors, vector addition, Closed Path Theorem, least common multiple, displacement, and symmetry.

Benefits of the exploration approach in developing mathematical ideas

In answering the proposed questions in these explorations, each question must first be understood, then a plan to solve each must be devised, implemented, evaluated, and revised as needed. As noted earlier, this involves a wide range of problem-solving techniques. The general problem-solving skills developed are applicable to a broad range of problems and can be used in a wide range of experiences.

In implementing these explorations at the secondary and undergraduate level, students are given the opportunity to review known mathematical concepts and learn new ones while exploring the applications of both. The application of these mathematical concepts in a new environment promotes an understanding of them, especially when students write procedures for the mathematical concepts. For example, to write a procedure to find the least common multiple of two numbers, students must understand the concept of least common multiple.

The three main benefits of explorations like these are a better understanding of the procedures, an enhanced understanding of the mathematical concepts, and improved problem-solving skills. The enhanced understanding of the mathematical ideas embedded in these poly procedures serves as a basis upon which new procedures can be written and new applications made. They, in turn, serve as a basis upon which new explorations can take place -- and exploration is the keyword of the Logo philosophy of education. This paper has shown that the Logo explorations approach can lead to understanding of important and advanced mathematical ideas, and thus is appropriate for advanced mathematics students.

References

Abelson, H., and diSessa, A. Turtle geometry: The computer as a medium for exploring mathematics. Cambridge, MA: MIT Press, 1980.

Billstein, R. Libeskind, S., and Lott, J.W. Logo: MIT Logo for the Apple. Reading, MA: Benjamin/Cummings, 1985.

Jane F. Kern
Francis A. Harvey

Educational Technology Program
College of Education
Lehigh University
Bethlehem, PA

Donald E. Kline
Francis A. Harvey

Educational Technology Program
College of Education
Lehigh University
Bethlehem, PA

Developing and Using Three-Dimensional Turtle Graphics Procedures for Apple and IBM Logo

The power and versatility of turtle graphics for programming and for teaching both programming concepts and mathematical concepts are accepted facts among computer users. As used in Pascal, COMAL, and LOGO, normal two-dimensional (2-D) turtle graphics have limitations. Living in a three dimensional (3-D) world and working with floor turtles naturally gives rise to the question, "Is it possible to extend the screen turtle beyond the confines of the two-dimensional surface, to simulate the realm of 3-D space?" This paper will describe a project which developed a complete set of 3-D turtle graphics procedures for MIT version 3.0 for Apples, and for IBM LOGO. The paper will discuss the mathematical foundations of the procedures developed, potential applications of those procedures, and the problems encountered in developing the procedures.

Abelson and diSessa, in their book Turtle Geometry (1980, Chapter 3), present the mathematical theory underlying the development of a generic 3-D turtle. The theory is based on treating the turtle's position and heading as vector quantities as it moves through space.

Since tools for manipulating vectors have not been built into the versionsof LOGO that were used, procedures for vector addition and scalar multiplication were developed. For example, the equationfor adding vectors:

$$v + w = ae_1 + be_2 + ce_3 + de_1 + fe_2 + ge_3 = (a + c)e_1 + (c + d)e_2 + (f + g)e_3$$

(were e_1, e_2, and e_3 are unit vectors) was translated into LOGO code as:

```
TO 3DADD :V1 :V2
   OUTPUT ( LIST ( FIRST :V1 ) + ( FIRST :V2 ) ( FIRST BUTFIRST :V1 ) +
      (FIRST BUTFIRST :V2 ) ( LAST :V1 ) + (LAST :V2))
END
```

The procedure 3DMULTIPLY, for scalar multiplication ($kv = k(ae_1 + be_2 + ce_3) = kae_1 + kbe_2 + kce_3$), was developed in a similar fashion.

These procedures were then used to describe the rotation of the turtle in 3-D space. In three dimensions, the turtle is no longer restricted to left and right rotations about an axis perpendicular to its plane. One must now consider movements about the three mutually perpendicular axes that make up the Cartesian coordinate system. These rotations will be referred to as ROLL (rotation about the turtle's longitudinal axis), PITCH (rotation about the axis in the turtle plane which is perpendicular to the turtle's heading), and YAW (rotation about the axis perpendicular to the turtle plane). Rotations are described relative to a reference vector associated with the turtle's heading, which is perpendicular to both the turtle's heading and the axis of rotation. Thus the general vector equation for rotating a vector **v** through an angle A is:

$$\text{Rotate } (\mathbf{v}, A) = (\text{Cos } A)\mathbf{v} + (\text{Sin } A)\text{Perp } \mathbf{v}$$

This equation is translated into the 3-D procedure:

```
TO 3DROTATE :VECTOR :PERPVECTOR :ANGLE
   MAKE "3DV1 3DMULTIPLY ( COS :ANGLE ) :VECTOR
   MAKE "3DV2 3DMULTIPLY ( SIN :ANGLE ) :PERPVECTOR
   OP 3DADD :3DV1 :3DV2
END
```

The 3DROTATE procedure is used to define procedures to rotate the turtle around each of its three axes. For example, the procedures YAW (equilivalent to the RIGHT primitive in normal 2-D LOGO), is defined as:

```
TO YAW :ANGLE
   MAKE "TEMP 3DROTATE :3DH :3DL :ANGLE
   MAKE "3DL 3DROTATE :3DL 3DMULTIPLY ( -1) : 3DH :ANGLE
   MAKE "3DH :TEMP
END
```

Three-D Turtle Geometry Procedures

Moving the turtle FORWARD in 3-D space can be visualized as moving it to the tip of a vector (P_{new}) beginning at the current turtle position ($\mathbf{P}_{old}$) and having a direction equal to the turtle's heading vector (H) with a length equal to the number of turtle steps. This can be written in vector notation as:

$$\mathbf{P} = \mathrm{P} + \mathrm{DISTANCE\ X\ H}$$

Representing the 3-D figure on a 2-D display surface is the final challange in developing a working 3-D system. This can be accomplished in two ways. Parallel projection shows the object without perspective. Perspective projection procedures are more difficult to write, since the procedures must keep track of the positions of both the point of view and of the turtle. Perspective projection procedures do, however, present a more realistic view of the object.

The complete 3-D procedure package, as developed, is memory efficient, requiring approximately 740 nodes. Thus it can be used with versions of LOGO with limited work space such as 64K LOGO versions. Speed of execution is slower than normal 2-D turtle graphics, but still acceptable in light of the larger number of calculations needed to carry out each command. SAVEPICT can be used to save the final form of any 3-D drawing for later use in demonstrations or in instructional programs.

Developing 3-D procedures provides an excellent programming exercise. Equally important, however, is the value of the process of developing the procedures for developing understanding of vector mathematics. The set of procedures developed can serve as very useful tools for teaching analytical thinking skills, and as visualization tools for producing graphical, three-dimensional displays in various subject areas (e.g., physics, chemistry, biology) which can be manipulated by the learner. They can also serve as tools for computer artists.

The 2-D turtle lead to the question of the possibility of a 3-D turtle. The 3-D turtle, in turn, leads to the question "Why not a 4-, 5-, 6-, or more-D turtle?"

References

Abelson, H., and diSessa, A. **Turtle geometry: the computer as a medium for exploring mathematics**. Cambridge, MA: MIT Press, 1980.

Exploring Language

Language as a domain for scientific inquiry

Language study makes considerable use of Logo's word/list processing tools, a programming territory long considered "hard" in Logo.

In some ways, this programming territory is genuinely harder than turtle moves. Turtle moves can be done one at a time, independently of one another, and their trace builds up on the screen just as if all the moves had been assembled first into a procedure and run "all at once." By contrast, manipulations of words and lists require us to plan the whole path that some piece of data will travel before we run any part of it.

However, entry into the list-processing world has been made unnecessarily mysterious by unfortunate choices of programming styles and inadequate metaphors for what is going on. We are accustomed to playing turtle, but we are less adept at picturing the complex interplay of procedures that output.

Fortunately, there are are consistent and relatively simple programming models, and many of the best language projects need little more than the simplest of these. Further, the use of iconic programming (see my paper with that title) helps people acquire and understand these models, and helps them design their own new procedures. In this paper and talk, I present some of the rationales for language-study, two of the simple programming models, and a small number of the very many explorations that are possible. (The long-awaited book by me and Wally Feurzeig, also titled *Exploring Language*, is finally in press! MIT Press, late this year or early 1987.)

Clearly, one motivation for language projects is the involvement that the student gets with language, from projects that exercise reading or spelling skills to projects that ask a student to derive some linguistic rule or model some complex linguistic process. Far more intriguing to me is the opportunity—a rare one—to do some real science! Students at any age before graduate school are rarely in a position to have free enough access to the data of a field to begin poking around, building hypotheses, and checking them out. So many experiments require fancy tools, expensive supplies, lots of time, and difficult techniques—radiotelescopes, swarms of laboratory animals, trips to the Olduvai Gorge, etc. But our own native language is a complex system worthy of study, and most of its data is right on the tip of our tongues, available even to young children. Without the computer, it is hard to play with linguistic theory, since it all dries up into paper rules and formulas, but with a computer the explorer can use hypothetico-deductive techniques (fancy for "think up a rule and then test it out") to inch up on the truth.

Some linguistic activities are so familiar in the Logo community that they are almost embarrassing to repeat, but I must begin with the most common one (reiterating a point I make in "Iconic Programming") to illustrate what I mean about the importance of picking a consistent model.

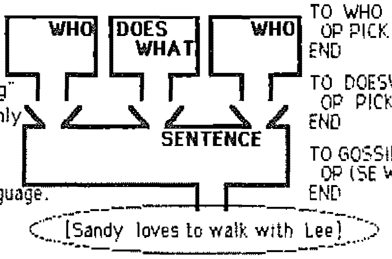
Compare these two programs for generating random sentences:

```
TO GOSSIP1                                          TO GOSSIP2
 PR (SE  PICK :PERSON  PICK :ACTION  PICK :PERSON)   OP (SE PERSON  ACTION  PERSON)
END                                                 END

TO SETUP                                            TO  PERSON
 MAKE "PERSON [Dale Dana Sandy Chris Lee]            OP PICK [Dale Dana Sandy Chris Lee]
 MAKE "ACTION [loves [sings to] hates kisses]       END
END
                                                    TO ACTION
                                                     OP PICK [loves [sings to] hates kisses]
                                                    END

?SETUP
?GOSSIP1                                            ?PR GOSSIP2
Dale sings to Chris                                 Dale sings to Chris
?GOSSIP1                                            ?PR GOSSIP2
Sandy loves Dana                                    Sandy loves Dana
```

Which is more easily learned? Put that way, the question is purely empirical, but there are theoretical considerations as well. Which is "simpler" depends upon your metric. GOSSIP2 is longer and requires PR in its invocation. But it is also a more consistent programming model, it requires less knowledge of Logo (OP instead of dots, quotes, and MAKE), it does not require the SETUP step before use. Perhaps the most important feature is that it generalizes so straightforwardly to more complex grammars, allowing, for example, compound or complex sentences to be composed from simple ones (e.g., GOSSIP2 itself) in precisely the same way that simple ones were composed from PERSON and ACTION.

Here is an iconic view of the structure I prefer.

By creating sources that do not always output the same thing, but rather PICK from a list of similar items, we can use "gluing" processors to recombine randomly selected chunks of a sentence or word, and, in that way, model and explore the structure of language.



```
TO  WHO
  OP PICK [Sandy Lee Dale ...]
END

TO  DOESWHAT
  OP  PICK  [[loves to...] [yells at]...]
END

TO GOSSIP
  OP (SE WHO DOESWHAT WHO)
END
```

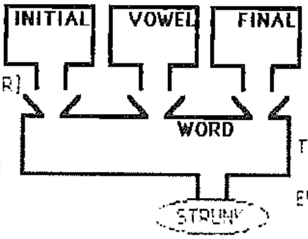
Because GOSSIP's surface behavior is the same as that of WHO and DOESWHAT —that is, all three procedures output chunks of text—it can be used as freely as the other two in composing yet more complex pieces of text. For example, consider the behavior or these procedures:

```
TO PREDICATE
 OP (SE DOESWHAT WHO)
END

TO QUESTION1
 OP (SE [WHO TOLD YOU THAT] GOSSIP [?])
END

TO GOSSIP3
 OP (SE WHO [TOLD ME THAT] GOSSIP)
END
```

```
TO COMPLEX
 OP (SE WHO [WHO] PREDICATE [ALSO] PREDICATE)
END

TO QUESTION2
 OP (SE [IS IT] WHO [WHO] PREDICATE [?])
END
```

Changing only the glue that we use but otherwise using the same structure, we can begin a new kind of exploration.

```
TO INITIAL
 OP PICK [STR SP TH CH P R]
END
TO FINAL
 OP PICK [NK NG G P ST L]
END
```

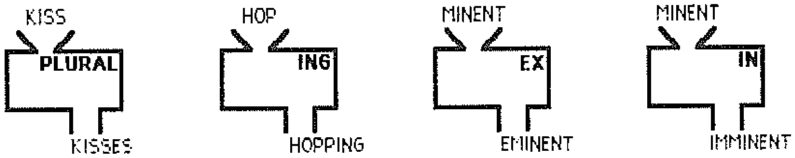


```
TO VOWEL
 OP PICK [A I U OO OI]
END
TO SYLLABLE
 OP (WORD INITIAL VOWEL FINAL)
END
```

For young students having difficulty sounding out words, programs of this kind are amusing nonsense word generators into which the children can learn to put their own choices of consonants, blends, and vowels. For high-school science students or college linguistics students, it might be a first cut at answering the question "What characterizes those collections of letters that look like an English word?" A scientifically interesting and sophisticated question is: What is a good metric for judging the relative goodness of two competing theories about what makes an English-looking word?

A second territory all too familiar to the Logo community is the task of developing a procedure that can suffix -s to a word, changing an English noun to its plural and changing a verb to third person singular, present tense. As with the GOSSIP programming, some early attention to what other linguistic directions such a project might lead may help us pick an optimally simple, flexible, consistent programming style. Suffixing suggests prefixing. Affixing in English suggest affixing in other languages. Follow-up projects might include conjugating a French verb, or designing a system for prefixing

Latin-origin roots (in English) with "in-" or "ex-." Here are some of the machines that might be developed for English.



Clearly, the construction of machines like these is in itself an exploration of spelling, morphology, and phonology. The use of machines like EX and IN also provides a route into etymology and semantics. How many times have you run across a confusion between *eminent* and *imminent* ? How does the meaning of the root and prefixes help recall which spelling goes with which meaning? If we create a machine that accepts words that end with -or or with -id and converts them to the other form (i.e., converting candor into candid and squalid into squalor), playing with it will expose some surprise pairs along with lots of non-pairs. We will get, for example, rapor and rapid, vapor and vapid, valor and valid, and rancor and rancid. Are all of these words? Are the pairs really related? How do experiments like these enrich the meanings of words?

Exploring Language with Logo

Michael Friendly
York University

[Keywords: Language microworlds; advanced topics; list processing]

To many people, language and mathematics are worlds apart. Mathematics is precise, abstract, and a product of formal instruction, whereas language is less orderly, more concrete, and ordinarily learned through experience and exposure. Yet they are both systems of rules and operations, patterns and transformations.

In this talk I will describe a variety of activities for exploring language patterns with Logo. In the same way that turtle geometry is a captivating gateway to intriguing discoveries in mathematics and art, and the links between them, the list processing capabilities of Logo can allow the child to work or play with language patterns at a wide variety of levels.

There are two central ideas: While the rules of language are largely implicit, we can make them explicit, like the rules of arithmetic, by describing a rule or operation of language with a Logo procedure. Second, a Logo procedure which defines a language rule, pattern, or operation is a working definition which can *compute* with words, sentences, and ideas (just as we compute with numbers), and we can use this effect to explore how language works.

I will illustrate these ideas in terms of a variety of Logo language activities which range from upper elementary to university level:

1. Word patterns: Plurals, MORE, Regular French verbs
2. Sentence patterns: Generating random sentences
3. Chomsky World: A one-command language for exploring generative grammar
   a. Sentence grammars
   b. Grammars for English-like words
   c. Grammars for pictures, stories, poems
4. Phrasebooks & Boxes (after M. Sharples): Packaging list processing for children.

Communications concerning this presentation:

Michael Friendly
Psychology Department
4700 Keele Street
Downsview, Ont.
CANADA M3J 1P3

Email: FRIENDLY@YORKVM1.BITNET

Girls, Computers, and Logo:

Viewpoints of a 6th Grade Girl, a Teacher, and a Counselor

Last year, during my presentation, at Logo 85 I made the statement that I knew girl participation with computers was generally much lower than boy participation and I wasn't sure why as that didn't seem to be true at my school with my students. I set out this year to learn more about girls vs boys in computing. Did I get an earful and they included but didn't stop with computers. I've been learning how the girls actually feel when they compete with the boys, be it real or imagined. Fifth and sixth grade girls perceive more of a problem than those of lesser grade levels. Some girls, of course, at all levels do not feel a problem exists.

This presentation will feature a demonstration by a vivacious sixth grade girl, a talented Logoist, who will also discuss her feelings about Logo and the "boy-girl" computer experience. This delightful young lady is quite verbal and has definite ideas of how boys and girls are treated differently in computing-----and beyond computing for that matter.

I've spent hours discussing these issues with Janet and other girls. She is an excellent spokesperson for them. These have been very enlightening conversations or perhaps I should say revelations. We have come a long way in our society, on paper. Have we really come that far in reality? As the young girls perceive it, no. I do believe there aren't equal opportunities for girls within our educational system. I do not believe this, in the majority of cases, is conscious sex discrimination. I also believe there are misconceptions by both girls and the significant adults in their lives. Perhaps, we, as educators and parents, are unconsciously sending incorrect signals.

This presentation will point out the feelings many young girls have as to the hurdles they

Beverly and Lee Cunningham
P.O. Box L
Treynor, Iowa 51575

Janet Szemplenski

face in gaining acceptance in the academic computer world. There are also true obstacles that we knowingly or unknowingly force them to overcome. We will then discuss how these hurdles, both perceived and real, are built, maintained, and how they can be removed.

When I talk to the boys about computer equality I wonder if I'm discussing the same subject. The boys have their own ideas on the subject. There appears to be many discrepancies between their ideas and those of the girls. Why do the boys and girls perceive the same situation differently? Are the boys so secure they don't notice the problems or are the girls supersensitive?

Just as there is no threshold, no ceiling in Logo itself, we must assure there are no obstacles in the path of any and all who wish to participate. We have come a long way towards girl-boy equality of opportunity in writing and by word of mouth. Many girls feel we have not progressed near as far in the real world as we would like to think. Until girls perceive a real change we have not been successful no matter what we write and what we say.

Dr. Seymour Papert succinctly states in Mindstorms, in respect to the child as an epistemologist, "children appropriate to their own use materials they find about them, most saliently the models and metaphors suggested by the surrounding culture". Does the surrounding culture of girls, in the way of models who are the significant people of their young lives, support in action and nonverbal communication that which we profess to be? If we, as parents, educators, loved ones, the general populace, really want the girls to participate in computing, is just saying it is OK to do so sufficient? Are the fathers and male educators supportive of the girls? Do these males show general cultural sympathy or practice overt positive action? Are the mothers and female educators supportive of girls as to individual needs and desires or only to the extent they perceive the culture will accept girls within a norm? What modeling do they personally present relative to technology and dealing with males? Now should you think "the lawyer is leading the witness", this is not a fishing expedition with negative expectations. We adult role models may be doing our best but how the girls perceive our intentions, actions, verbal and nonverbal communications is the true indicator of our success or of our need to do better.

The manager of a large retail computer store in a metropolitan area who has been involved in retail sales to parents for several years responds; Young parents seem, as a rule, as interested in a daughter as a son, but the boys are more aggressive in "pushing" their parents for a purchase. Generally, parents are together in making a purchase for a son but sometimes it is only the mother if the purchase is for a daughter. If it is grandparents making the purchase for a grandchild, it is the grandfather who plays the main role in the purchase and the computer is nearly always for a GRANDSON, not a GRANDDAUGHTER. Does this mean we merely need to wait another generation or two for this societal problem to disappear? Time alone would be most unlikely to achieve the desired result and more important why should many talented girls be left out of this technological age today. They need to be involved and we need them involved.

During the presentation, as you hear the viewpoints of Janet, a teacher, and a counselor, you will be given a lot of food for thought, some possible solutions, and some do's and don'ts. Audience comments will be welcomed. Perhaps, we can all leave Logo 86 with a better understanding of how many girls feel when they want to enter computing at school and what we, as individuals, can do to encourage girls to participate in computing and help them feel as accepted as their male counterparts. Let's all be a part of a constructive solution.

THE MANY DIMENSIONS OF LOGO

Logo: Logo - Turtle Geometry; Logo - list processing; Logo - an open-ended medium for teaching/learning; Logo - a medium for technological change; Logo - a philosophy of educational change. All of these dimensions of Logo can be viewed as separate entities or as an integrated whole. During the "Creative Use of Computers" project, carried out at Queen's University during 1982-1984, thirteen teachers and five researchers explored all of these dimensions of Logo. The outcome of these explorations are well documented and illustrated in the final research report (Carmichael, et al, 1985). Findings that bear on the last three dimensions warrant particular attention since they can contribute to a greater understanding of what has transpired to date within the Logo movement and since they can offer possible guidelines for future directions. Above all, they remind us that adults are co-learners in this movement and that they, like the children, need environments which encourage exploration and learning and which offer opportunities for examining the process of their own learning.

Logo: As Medium or Philosophy of Educational Change

A persistent dilemma, throughout the project, in assessing the feasibility of using Logo as a vehicle for bringing about fundamental changes in the teaching/learning environment was the temptation to evaluate or make judgments prematurely. To counteract this temptation, it became absolutely critical to become sensitive to the underlying process of transformation and to acknowledge the various stages in this process.

Since the introduction of Logo also meant the introduction of a new technology, the first stage of the transformation process related to accommodating the computer technology (Logo) itself. At this stage teachers were preoccupied with the following two questions:

1. How can Logo be justified within the confines of the existing curriculum or how can the technology (Logo) serve in delivering the existing curriculum?
2. How can the technology (Logo) improve the delivery of the existing curriculum.

Even though these questions did not challenge the underlying ideology or teaching practices, they still generated fears or concerns that needed to be aired and examined before teachers were set free to find possible answers to them. Some of these were:

1. Fear of the technology: Is it potentially dehumanizing?
2. Fear of not knowing (enough) about computers, Logo or other software.
3. Concern over having to cope with many new developments (new teaching assignments, special education, etc.) in the normal course of teaching.

4. Concern over lack of support from principal and colleagues.
5. Concern over lack of recognition in any formal way for their work with computers.
6. Concern over limited access to computers: Is it worth all the effort?

Only after the teachers were able to resolve these issues to their satisfaction were they able to move to the next stage and explore the question:

> Can the technology (Logo) offer a new way of teaching/learning and can it broaden or redifine what is currently being taught?

In order to look for answers to this question, teachers needed a safe and accepting environment since it demanded experimentation in the reaches of the unknown. This stage generated many more fears and concern:

1. Fear of the unknown.
2. Fear of making mistakes; looking stupid; of not having all the answers ready.
3. Fear of loosing control.
4. Fear of having to cope with an unfamiliar teaching style or process.
5. Concern over accountability.
6. Concern that the time required to explore new alternatives will cut into precious private time.
7. Concern over taking too much time from the regular curriculum.
8. Concern over peer reaction in the event they should interfer with "their" curriculum.
9. Concern over classroom management.

Once teachers had resolved these issues they were further confronted by extrinsic factors that influenced to what extent they were able to actualize the new vision in the classroom. Some of these factors were:

1. The degree of tangible support that the teacher received from the princial and/or colleagues.
2. The grade level; the number of students and the range of abilities among students in the class.
3. The number of computers available; the duration of access and whether they were located in the classroom or in a laboratory.
4. The amount and type of assistance available from volunteers, students or other support staff.
5. The degree of flexibility in reorganizing the class routine: time-table; lab-schedule; subject-specialization.
6. The demands placed on teachers by other teachers or administrators inside and outside the school.

7. The familiarity the teacher had with teaching the routine curriculum for his/her class.
8. The type and quality of computer courses available to the teacher.

There is strong evidence that if Logo is to continue to serve as a medium for change that all of these potential inhibitors need to be dealt with collectively and explicitly rather than left to the individual or left smoldering.

Logo: A Rich Open-ended Medium For Teaching/Learning

All of the teachers in the project were eager to explore a new dimension of teaching/learning. They explored many alternatives and they collectively discovered that Logo, even in the form of elementary Turtle Geometry, simple list processing and its music utility, can become a rich medium for teaching/learning. They used Logo in any one or several of the following ways:

1. As a means to teach a programming language.
2. As a means to explore alternative solutions to a given problem.
3. As a medium for exploring several fundamental concepts in mathematics by using Turtle Geometry as a mathematics laboratory.
4. As a medium for facilitating language development by using its editor as a simplified wordprocessor or for exploring language in a dynamic sense through interactive quizzes or stories that utilize the list processing capabilties of Logo.
5. As a medium for exploring simple concepts in music.
6. As a medium for exploring art, especially geometric patterns and designs.
7. As a medium for integrating several of the above topics.
8. As a medium for exploring problem solving in a variety of contexts: mathematics, language, art, personal, social, societal.
9. As a means to encourage initiative, free inquiry and undirected experimentation.
10. As a means of extending social awareness by sharing the Logo experience with emotionally and mentally handicapped children.
11. As a means to sensitize students to elements of good teaching by giving them opportunities to teach others: peers, younger students or adults.
12. As a medium for designing a suitable computer language for very young children.
13. As an alternative classroom environment over which students have control and are free from judgment or evaluation.

Although the first eight uses of Logo can be formulated within the context of the traditional teaching/learning ideology, these teachers were able to demonstrate that very fundamental changes in teaching and learning strategies can take place even in the regular classroom, **provided the necessary support is given.** They also have shown that Logo can initiate and facilitate such fundamental changes **without immediately challenging the entire curriculum or school organization,** yet it can still be used to transcend the confines of the computer language and the existing curriculum.

Reference: Carmichael, H.W.; Burnett, J.D.; Higginson, W.C.; Moore, B.G. and Pollard, P.J. (1985); Computers, Children and Classrooms: A Multi-Site Evaluation of the Creative Uses of Computers by Elementary School Children; Ontario Ministry of Education, Publication Services, 5th Floor, 880 Bay St., Toronto, Ontario, Canada, M7A 1N8; $20.00 (Can.); 446 pages.

For further information, please contact:

Hilda Carmichael
206 Victoria St.
Kingston, Ontario
Canada
K7L 3Y8

PRESENTATION

THE REALITIES OF IMPLEMENTING A LOGO CURRICULUM IN TEXAS PUBLIC SCHOOLS

JUOY ROBBINS, INSTRUCTIONAL COMPUTING CONSULTANT
CINDY PARKS, FIFTH GRADE TEACHER
CARROLLTON-FARMERS BRANCH ISD
1445 N. PERRY RD., CARROLLTON, TEXAS 75006

The conceptual basis for Logo stands in the experiential process of learning in an interactive, supportive environment. This process is dependent to a large extent on the motivation provided to the student by the learning process itself. When access to the learning environment is denied by the restrictions of time, equipment, and educational policies which commonly occur in public school systems, there is a tendency to respond that Logo is not appropriate and cannot work in the public school environment.

This response is the equivalent of throwing out the baby with the bath water. We were not willing to exclude Logo from our district because the ivory tower theories did not fit the reality of today's public school education delivery systems. Working within the constraints of public school policy and the State of Texas mandated educational requirements, we at Carrollton-Farmers Branch Independent School District have been able to implement a Logo educational program for all fourth and fifth grade students in the district.

This presentation examines the process that we went through in implementing the program. It identifies the restrictions and constraints that were encountered and how we were able to resolve these problems. In this arena the presentation deals with the conceptual aspects of implementing a curriculum for managing Logo instruction. This includes not only the teaching materials, but also the teacher training, computer hardware and software availability, and the logistics of student involvement.

In this presentation, each of the following questions are raised and discussed:

Constraints and Restrictions:

Computer equipment was already installed in all elementary schools in the district--a lab of 12 Commodore 64s in each elementary school. Given Texas mandated education requirements, students were only left with one hour per week in the Logo lab. District policies dictated that students would have to be taught by their own teacher--it was not possible to have a Logo teaching specialist. Very few of the teachers had any training in Logo and an appropriate curriculum was not available.

Making a Start:

In order to evaluate alternatives we began by seeking expert help. Judy Robbins attended the MIT Logo 84 conference. Based on that experience a foundation curriculum (Turtle Teacher) was developed. A Logo university course was offered in the district to train teachers. Graduates of this course were enlisted to explore implementation ideas. A "brainstorming" session with professional consultation, created the foundation of the program.

Curriculum Development:

A teacher Task Force was created for establishing sequenced learner objectives which provided experiential learning. This framework was required to ensure consistency of the curriculum throughout the district. With this framework available, we began writing lesson modules for the objectives.

The lesson modules were scripted as if they were being taught to elementary students following Madeline Hunter's "Effective Teaching Model." Also included in these modules were worksheets, transparency masters and extension activities. Extension activities provide further exploration for faster students, so the teacher may address the special learning needs found in the regular classroom. Activities were designed using a partner concept to insure equity since students are required to share a computer.

Implementation:

Additional teacher training programs were implemented making use of both internal district training and a graduate Logo course. The graduate course could be counted by the teacher for state approved career ladder advancement. Every 4th and 5th grade classroom teacher in the district completed either the university Logo course with a follow-up district training in the use of the curriculum materials or the equivalent training within the district. Funds were provided for substitutes to allow teachers to take the district training.

A pilot program was initiated in an elementary school to test the lesson modules. Based on the results of the pilot program, adjustments and corrections to the lesson modules are made.
The Logo curriculum has been implemented in all the 4th and 5th grade classes in the district.

Program Results:

The Logo program is receiving praise from students, parents, teachers and administrators. Students are extremely enthusiastic and look forward to their one hour per week in the computer labs. Teachers love it--the scripted lesson plans gives them a wide range of support. They can follow the scripted model or use it as a framework for teaching. The parents are enrolled by the students and like the idea that their children are solving problems using a computer. Many of the parents are purchasing computers and Logo for their children to use at home.

Conclusions:

Logo can work in public schools in this country even when it is not presented in the environment originally envisioned in theory. We have shown that Logo has a place in public schools even when the students share the computers and can only spend one hour per week working with the computer. Teachers working within a structured curriculum that emphasizes Logo as a problem solving tool can create a environment where learning occurs through the student's interactive experience. Because we are teaching Logo in the public school, many children have begun to use it in the home. In this way, the public school has become the vehicle which inspires the independent creative environment originally envisioned.

Logo and Logoism

Denis Donnelly
Department of Physics
Siena College
Loudonville, NY 12211
518-783-2440

A not uncommon cycle, when a new technology is being introduced into the educational system, is a successful seminal beginning, the hope that the system will solve almost intractable problems, and numerous revolutionary claims about the role of the new system in the future. As time proceeds, problems do not yield to the onslaught, complexity is appreciated, claims are scaled back, and predictions of future accomplishments are more cautious.

The educational process has experienced this sequence of events on several occasions. After the successful introduction of television, certain visionaries made extraordinary claims about the impact of this technology. One scenario was that the nation's best scholars would present lectures in their areas of expertise and every school or college in the nation would then have available regular presentations by scholars of the first rank. The lesser academics would become more like teaching assistants, relegated to answering questions. What institution would ask an unknown Mr. X to lecture if Mr. Nobel Prize was available? Of course, the outcome was somewhat different. The hardware worked but the programming did not materialize as initially envisioned. Before the limitations were realized, some schools made massive investments in equipment which was never fully utilized. Television does have a role but a vastly different one from those first projections.

However, the initial underlying concept, the idea that something would come along and solve the problem of education, remained. Some people seem to like the idea of <u>a</u> method, <u>a</u> solution, <u>a</u> fix. That there is no single method or solution or fix is rarely considered. Some of the early history of the field of artificial intelligence demonstrates the same single-line approach. The project known as general problem solver (GPS) would fit into that same category - one marvellous answer. Yet GPS has, for all intents and purposes, been abandoned. Despite its title GPS was not really very general; it was really only suited to mathematical/logical problems such as playing chess and cryptarithmetic. It dealt only with closed questions.

What of Logo? The claims are of the same general "this-is-it" category. Logo is a "computer-based learning environment"(1); "Logo is a language for learning how to think"(2); "Logo is tuned for interesting applications"(2); "If we can dispel the delusion that learning about computers should be an activity of fiddling with array indexes and worrying about whether X is an integer or a real number, we can begin to focus on programming as a source of ideas"(1); "Logo is easy enough for anyone to use, but it is powerful enough for any project"(2); Logo is "a door into the territory of the computer as an object for intellectual exploration"(2); Logo is "a philosophy of education". Logo would revolutionize the curriculum; learning would never be the same.

Yet the fact is the curriculum has not changed significantly. Logo has not revolutionized the curriculum. Preliminary studies show that students having had Logo experience are no better at solving problems outside the immediate Logo environment than the rest of the student population. What is the emperor wearing?

Logo is in many respects a superior language, having many desirable features - procedural, interactive, recursive, has list processing, is not typed, and is extensible. Yet that may not be what is needed for the ideal learning environment, as neither these language features nor those of any other language lead broadly toward the two essential features of the educational process necessary for learning to think - writing and problem solving.

List processing is an elegant feature and provides the user with powerful manipulative capability. While Logo is vastly superior to BASIC or other popular languages when it comes to list processing operations, the student has no real need for such processing capabilities. But in terms of words, what is essential for the student? No programming at all; writing is what the curriculum should require. Writing is what is needed. What the student needs is a simple yet reasonably powerful word processor. The student should write and revise frequently. Any list process programming, however elegant, clever, or educational is peripheral to the central task of writing.

Turtle geometry is magnificent. The text "Turtle Geometry" by Harold Abelson and Andrea diSessa (3) is masterful and a fine example of pedagogy. However, if one looks carefully at what is included in the text - feedback, growth, and form, vector methods in turtle geometry, topology of turtle paths, turtle escapes the plane, exploring the cube, etc. - and then asks oneself which of these topics are essential to the curriculum, which are central or fundamental, the answer, I believe, is close to none. As beautiful and elegant as the approach is, it is also almost totally peripheral. It does not get at what are generally considered to be the more essential problems. For example, it is yet to be shown that high school mathematics - elementary algebra, Euclidean geometry (statement-reason proofs), and trigonometry - is better grasped by those students in a Logo environment.

Part of the problem is that there is no problem; there are many problems. Think back on the measurement of IQ. For years much was made of IQ testing and IQ scores as a measure of intelligence. It is now generally agreed that intelligence is multifaceted and no single measurement of IQ could provide a measure of "intelligence". Mind as envisioned by Marvin Minsky is a "society" with thousands of agents combining in a complex system, where the languages of the subparts are not even the same. Minsky suggests that "in order to comprehend the brain, we may have to learn to use several thousand organizational concepts "(4). It would be disingenuous to suggest that any computer language or technology currently imagined is best suited to interface with something as complex as that society.

1. Harold Abelson, Byte 7, 88 (1982).
2. Brian Harvey, Byte 7, 163 (1982).
3. Harold Abelson and Andrea A. diSessa, Turtle Geometry (MIT Press, Cambridge, 1980).
4. Marvin Minsky, The Society of Mind, to be published.

Getting to Know Logo: A Magical Experience

Greg Robertson
Junior at Wooster High School
Sponsored by: Sharon Burrowes

My first experience with computers was anything but pleasant. I was sitting in math class with my friends when the teacher announced that over the course of the year, my friends and I were going to learn to program a computer. At the time, I had no fears of computers and was anxious to use one. After the teacher began explaining Basic to my class, I quickly changed my attitude toward computers. I was too "cool" at the time to ask any questions and left class each day farther behind my classroom peers. We only had access to one computer, thus computer time was very limited. I never did understand Basic very well and I found it necessary to copy the ideas used in my friends' programs to keep up in the course.

When it came time to enter high school, I decided that I must take a computer course to compete in the future job market. The only course offered for a beginner like me was Introduction to Computers. I didn't want to take the course, but I felt that I must do it to keep up with everyone else. I think it was the first day of class when I found that we were going to be learning Logo. I could hardly believe my ears. If I recalled correctly Logo was the "baby's language" that my little brother was learning in elementary school. I didn't read the course description before I enrolled and assumed that a high school course would deal with something more sophisticated than Logo.

I was disappointed further when I learned that we weren't even going to begin the course with Logo. I figured that anything we were going to do that was easier than Logo had to be busy work, and indeed some of it was. When we first saw Logo, we had to do our initials in immediate mode; and though I had fun, I became more skeptical than before. Since all the commands that I knew were graphics commands, I figured that I was going to spend an entire semester drawing pictures. When procedures were introduced, I began to like Logo a little more. Before procedures, Logo seemed to be totally useless except as a glorified sketch pad.

After about the first week of Logo, the class saw the video tape Talking Turtle which featured Semour Papert and children using Logo. I couldn't believe it. I watched children not even half my age drawing houses and flowers. I could barely draw my initials. Now I knew that I was in trouble. How could I possibly compete for jobs with children like that running around? Luckily, the teacher was there to encourage everyone to keep at it. She also said that my classmates and I would be able to do the same thing in a short time. I felt like the Yoga students who are shown what they will be doing in a short time. My inferiority complex returned, and I began to contemplate dropping the course.

The main reason why I didn't drop the course was that I was extremely curious about Logo, and I even liked it a little. After about the fourth program, I began to feel good about myself. I had a disk with programs on it that I had made without the help of anyone else. As the course continued, both in class and out, I became increasingly proud of what I could do. I also began to love that language that only a month ago I had hated. I spent all of my free time in the computer lab working with Logo. When the course was over, I even asked the teacher if she would give me assignments to do during the next semester since I wanted to work with Logo more.

My reason for loving Logo so much is that it presents insight into one's self. I put a little of myself into every project that I did. I took so much pride in my computer projects that I ceased to think of working with Logo as an assignment and began thinking of it as recreation. In the lab around me I saw others experiencing that same sense of accomplishment. What interested me at the time, was the fact that everyone seemed to have a distinct style and ability. Even something as simple as a greeting card was so different for everyone that we delighted in looking at each others projects.

Shortly after my talk with the teacher, she approached me about an independent study course. If I was interested in this course, I could sign up and I would even get grades and credit for the course. I didn't really care about credit; all that was important was that I was going to get to continue my interaction with Logo. During the course, I played with some new and interesting parts of Logo. One thing that was particularly interesting was "broken Logo". "Broken Logo" is merely pretending that some primitives in Logo are broken, and that you the programmer must fix them. To do this, the programmer can use other primitives to create a procedure to do the required task, but the broken primitive can't be used. Another thing that the class dealt with was an independent project. Remembering the pie-graph program on my neighbor's I.B.M., I wanted to write one of my own. Using some of Logo's special primitives, like property lists, I was soon able to make pie-graphs on my own program. After my independant study in Logo, I felt that there wasn't anything that I couldn't tell Logo how to do. I wished that I could spend my summer writing Logo programs. I even offered to adopt a school computer for the summer.

I would say that Logo is one of the most exiting things that has happened in my life, and I think that everyone can benefit from it. From my point of view, there are several things that I think it is important for teachers to remember. First, it is important to force structure on students. In the beginning, structure made little sense to me; however, when my programs got more sophisticated, I was glad that they were so easy to understand. Thus, good structure makes better programmers. Also it is important to realize that all students are very different. This means that computer programming assignments should be broad enough that everyone can enjoy them. Just as everyone is not a talented musician or an artist, everyone takes a different approach to programming. Lastly, and perhaps most importantly, remember that Logo is a tool to help you and your students to grow and expand your horizens. Logo is useful, but it is also fun and should always be approached as such.

For further information, please contact:

Greg Robertson
2482 Townsend Dr.
Wooster, Oh 44691

SHOULDN'T REVOLUTIONS TURN THINGS AROUND?

> You say you want a revolution
> Well you know
> We all want to change the world
> ...
> You tell me it's the institution
> Well you know
> You better free your mind instead
>
> John Lennon, "Revolution 1"

Come the revolution education is not just supposed to be different; it is supposed to be better. It isn't and what is more troubling it is not much better in our community, that of teachers, learners and researchers who use Logo. Why is there still so much education as usual even among us? Whatever happened to the revolution indeed?

This inertia is largely the result of two key mistakes: one is what we are all too often doing and the other is what we are even more often failing to do. On the one hand, being in the grip of what Papert has called "the first instinct of educators," we are using new technologies to perpetuate old methods of instruction.[1] On the other hand we are not keeping in mind that Logo is also, and in a certain sense primarily, a philosophy of education[2] and hence we are forgetting the corollary that the Logo revolution is a revolution of ideas, not technology.[3]

Were we to do philosophy of education more frequently, more thoroughly and more carefully, it would go a long way toward liberating us from our first instinct. You might say that mindstorming would help overcome footdragging. This presentation is an outgrowth of a little mindstorming about the educational ideas of two great philosophers: the Socrates of the Platonic dialogues, in particular Meno and Protagoras, and John Dewey. I argue that Logo properly used is Socratic and Deweyan in nature and consequently, that incorporating the insights of and sharpening the skills championed by these philosophers of education will help bring about the revolution we all want.

ENDNOTES

[1] Seymour Papert, "New Cultures from New Technologies," BYTE, September 1980, p. 230

[2] Seymour Papert, Mindstorms (New York City, N.Y.: Basic Books,1980), p. 217

[3] Ibid., p. 186

William McCurdy
704 Shulos Miyashita
Miyashita 3-Chome
Asahikawa, Hokkaido
Japan 070

LOGO 86 SPONSORS

Apple Computer, Inc.

IBM

LEGO Systems Inc.

Logo Computer Systems Inc.

Terrapin, Inc.

APPLE COMPUTER, INC.

Apple Computer pioneered the use of personal computers in education and is the leading supplier of computers to schools. In its short history, Apple has helped define the role of personal computers in education, has educated people to the potential of these machines and has developed systems and software in direct response to customer need.

Apple's goal is to help motivate America to take an aggressive lead in technology and technological education. As a leading manufacturer of personal computers, Apple strives to promote awareness of the benefits of personal computers in education and the importance of preparing students and educators alike for a technological future.

"As a company, we have an unshakable belief that personal computers can greatly enrich the capacity for students to learn, to grow and to achieve, " says John Sculley, President and CEO of Apple Computer. "We believe that personal computers can help develop those leaders of tomorrow, and help the nation make its transition into a high technology society.

It is not enough to use computers for things like computer literacy and drill and practice. Computers should be used as intellectual tools that will help students learn concepts, develop their intuition and expand their creative abilities. They must be used to help the leaders of tomorrow understand and participate in an increasingly complex world."

Since its incorporation in 1977, Apple's commitment to education has led to the implementation of programs that have had direct impact on the educational community. Through such programs as the Apple Education Affairs Program (initiated in 1979 as the Apple Education Foundation), Apple has supported the development of new methods of learning and teaching through the use of personal computers. Our donations of equipment and software have supported projects that have created model education software, and projects that foster the use of computers in the classroom as intellectual tools. The latest grant cycle was targeted at supporting institutions with students who are disabled, economically disadvantated, represent ethnic and linguistic minorities, and girls and young women studying science and mathematics.

Several projects in each of the Foundation's grant cycles have been based on the use of Apple's Logo products. This is consistent with our philosophy of technology as a tool, and the Logo projects have provided an exciting example of our vision in today's world.

Our jobs are not over, however. Together, we at Apple and you as educators are facing a major transformation of the education system. Every two and a half years, the amount of information in the world doubles. By the time today's primary students graduate from high school, the amount of information they are going to have to cope with will have increased 16 times . We have to prepare students for that world by giving them thinking and analytical skills, and do it in a way that allows them to transport those skills across very diverse areas of information. Your projects and those you will hear about during these proceedings are exciting forerunners of the future. Your commitments and ours will serve to bring a new generation of students into leadership positions.

We welcome you to Logo '86. Together we will see the next generation - today!

TERRAPIN, INC.

Current Activities and Future Developments

At Terrapin, we are constantly striving to be the best at what we do. Terrapin introduced the first version of Logo for the Apple computer and has been improving and upgrading it ever since. We have also developed versions of Logo for the Commodore 64 and the Commodore Plus 4.

Terrapin is not only committed to making more and better Logos for the most popular machines, Terrapin is committed to education. We found that teachers generally did not have sufficient time to implement Logo in their classroom. Therefore Terrapin created the Logo Works series to take some of the pressure off of teachers. These books act as a core around which a Logo curriculum can be easily built to supplement and extend standard subject areas.

In the future, Terrapin will be creating subject specific disk and print material. These materials will allow much greater exploration of a subject than a general treatment of Logo could. The projects will provide opportunities for students to apply and experiment with concepts they have learned in the classroom.

Last year Terrapin upgraded its Logo for the Apple II series to version 3.0. The features incorporated in the 3.0 version have been well received. Most important of these features is that both a 64K and 128K version comes on the same disk. The computer boots automatically to the correct version. Both versions use DOS 3.3 so the files are completely compatible. In addition, all other features from the editor commands to the graphics are identical between the 64K and 128K versions so users will not be confused by having to switch between two different syntaxes. Another nice feature is that the time to boot Logo has been reduced to under 7 seconds.

Commodore 64 Logo has been extremely popular. This Logo features sprite and music capabilities. It has the same extensive documentation of the Apple version, and the syntax is nearly indentical. Commodore Logo can also run on the Commodore 128 computer. A different version of Logo is available in cartridge form for the Commodore Plus 4 computer. The Logo version for this powerful but very inexpensive computer has almost twice the amount of workspace as the Commodore 64 version, and the booting time is less than a second.

* **Terrapaks** give schools a less expensive way to purchase multiple copies. They have been very popular in school lab settings. The Terrapaks come with one set of documentation and 10 or 20 Logo disks.

* **Networked Logo** allows networking of Logo on Corvus hard disk systems.

* **The Utilities II package** provides 29 useful utilities and demonstration programs, including Imagewriter and Color Plotter drivers.

* **A new Demonstration Package** is available for qualified persons desiring a demo program of the Terrapin Logo Language for workshops, conferences, or school board review.

* **Music Logo** was released in February of 1986. It was developed in conjunction with Jeanne Bamberger of MIT and has been extensively tested and revised over the last three years. Music Logo is intended for use by a wide spectrum of music enthusiasts; from the elementary school student with little or no musical training up through the professional songwriter or musicologist. Terrapin expects Music Logo will stimulate the development of innovative ideas for teaching music in the music curriculum of primary grades through university music theory courses.

* **The Logo Works series** resulted from Terrapin listening to numerous requests for quality educational support materials. These materials are designed to assist novice as well as advanced teachers in their instruction of Logo, and their use of Logo to explore other subject areas.

The first in this series is **Logo Works: Lessons in Logo.** It is designed to integrate Logo into the 4th to 8th grade geometry and computer literacy curriculum. It was created by educators and computer coordinators from Chapel Hill-Carrboro City Schools and tested for 2 years.

The second in the series is **The Logo Project Book: Exploring Words and Lists.** The Logo Project Book explores language, patterns, and mathematics using Logo's powerful list processing capabilities. Alison Birch developed the projects during her 7 years of teaching with Logo. The book treats "words and lists" with the same playful engaging accessibility of the best of the Logo turtle graphics material.

The Logo Works series is currently available for a free 30-day trial for those persons interested in previewing the materials.

Future plans

Currently Terrapin is developing Logo based applications in the mathematics area covering a wide spectrum of specific subject areas. Some of these include trigonometry, probability, geometry, and functions.

These new products will be primarily software based tools and will be accompanied by directive teacher support materials. The topics will mainly be those covered in middle school through 12th grade math. The major brands of microcomputers and versions of Logo will be supported. Terrapin has received numerous requests for such materials and is enthusiastic about the innovative approach being used to develop the materials.

Terrapin also continues to pursue development of its highly regarded version of the Logo language for other widely used microcomputers.

Terrapin prides itself on its customer support. If you have any questions about any of our products, please write or call

TERRAPIN, INC.
222 Third Street
Cambridge, MA 02142
(617) 492-8816

Logo Computer Systems Inc.

Thinking Tools for All Peoples of the World

Since 1980, **LCSI** (Logo Computer Systems Inc.) has brought together some of the world's finest computer programmers, systems engineers, educators and technical writers to develop the Logo computer language for a wide range of microcomputers. The Logo computer language was originally developed at MIT by Dr. Seymour Papert and his team of researchers. Dr. Papert is the Chairman of the Board of LCSI as well as its Senior Techical Advisor for new products.

In 1982 and 1983, two **LCSI** products (Apple® Logo and Atari® Logo) were awarded **The Best Microcomputer Software Of The Year Award** by the Learning Periodicals Group . BYTE Magazine has called **LCSI** software, "the Cadillac of Logos". As well, in 1983, Apple's celebrated "Kids Can't Wait Program" which put microcomputers in over 9,000 California schools chose **LCSI**'s version of Logo as the programming language to be installed on each computer. In 1986, **LCSI**'s Logo for the Macintosh earned the Best Microcomputer Software of the Year award by *Classroom Computer Learning.*

LCSI has enjoyed numerous successes since its founding by a handful of dedicated professionals. Today, despite the growth experienced by the Company, it has deliberately kept the responsiveness and energy of a small company. The **LCSI** family spirit remains very much prevalent.

Starting in 1986, **LCSI** is re-dedicating its effort towards the development of useful educational tools to enhance the learning process, and directly providing these to all schools through an innovative site licensing program. The flagship product in this program is LogoWriter™. This innovative integration of a programmable text processing microworld within an advanced Logo has already won critical acclaim and a fast-growing user base. Also, the abundant teacher and student support materials have set new standards for educational products.

Noted for superior documentation and software design, the products of **LCSI** have become the **International Standard** for excellence in educational software. To date, **LCSI** has developed over 20 versions of Logo which span 8 spoken languages and 13 different brands of computers, and this is only the beginning...

Apple®Logo
Apple®Logo (French Edition)
Apple®Logo//
Apple®Logo// (French Edition)
Atari®Logo
FM™Logo (Fujitsu FM-7 computers)
IBM® Logo
Logo for the MacIntosh™
Logo for the Professional™
300 Series
Logo Learner (for the IBM PCjr)
MAC LOGO™
MO5™Logo(Thomson-Brandt Computers)

MSX™Logo (Arabic,Dutch,English,French,
German,Italian,Japanese,Spanish)
Nabu™Logo
NEC™Logo 66
NEC™Logo 80
Sinclair™ZX Spectrum™Logo 1
Smart Logo™(Coleco-Adam Logo computer)
Sprite Logo
TO7™ Logo (Thomson Brandt- computer)
LogoWriter™ (for the Apple //e, Apple//c,
IBM PCjr, and IBM PC and compatibles)

Apple® is a registerd trademark of Apple Computer , Inc. Atari® is a registered trademark of Atari Inc. FM™ is a trademark of Fujitsu Limited. IBM® is a registered trademark of International Business Machine Corporation. Macintosh™ is a trademark licensed to Apple Computer, Inc. Professional™ is a trademark of Digital Equipment Corporation. MAC LOGO and LogoWriter™ are trademarks of Logo Computer Systems Inc. MO5™ and TO7™ are trademarks of Thomson Grand Public. MSX™ is a trademark of Microsoft Corporation Nabu is a trademark of Nabu Manufacturing. NEC™ is a trademark of NEC Corporation. Sinclair™ and ZX Spectrum™ are trademarks of Sinclair Research Ltd. Smart Logo™ is a trademark of Coleco Industries.

LOGO 86 Exhibitors

Coral Software Corporation

Creative Publications

Harvard Associates

Heath/Collamore

Holt, Rinehart & Winston Inc.

LOGO Publications

National Logo Exchange

Nueva Learning Center

Webster Division, McGraw Hill Book Company